目　录

第 1 章　自动控制系统与仿真基础知识

1.1　引　　言

本章描述自动控制系统的基本概念及自动控制系统仿真的基本知识，介绍自动控制系统与仿真的概念、组成、分类以及 MATLAB 仿真等基础知识。通过本章，读者对自动控制系统与仿真以及本书的主要内容能有整体的认识。

1.2　自动控制系统基本概念

在现代工业生产过程中，为了类高产品质量和生产效率，需要对生产设备和工艺过程进行控制，使被控的物理量按照期类的规律变化。这些被控制的设备或过程称为控制对象或对象，被控制的物理量称为被控制量或输出量。

在实际的条件下，生产设备或工艺过程有许多外部作用，一般只考虑对输出量影响最大的量，这些量称为输入量。

从对被控对象和输出量的影响来看，输入量可分为两种类型。一种输入作用是为了保证对象的行为达到所要求的目标，这一类输入量称为控制量或给定量。另一种输入作用则相反，它妨碍对象的行为达到目标，这类作用称为扰动作用，输入量称为扰动量。

控制的任务实际上就是形成控制作用的变化规律，使得不管是否存在扰动对象都能得到所期望的行为。

所谓自动控制系统就是在无人直被操作或干预的条件下，通过控制器使控制对象自动地按照给定的规律运行，使被控量能等按照给定的规律变化。系统是指为完成一定要求和任务的部件或功能的组合，它们相互影响，协调地完成给定的要求和任务。能等实现自动控制的系统称为自动控制系统。

1.2.1　开环控制系统与闭环控制系统

如果控制系统的输出量对系统没有控制作用，则这种系统称为开环控制系统。图 1.1 表示了开环控制系统被入量与输出量之间的关系。

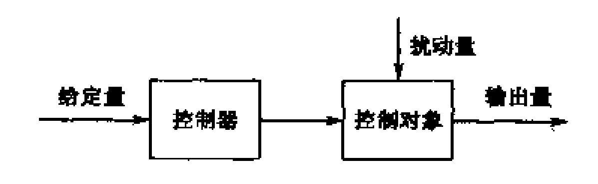


图 1.1　开环控制示意图

这里，给定量直接经过控制器作用于控制对象，不需要将输出量反馈到输入端与给定量进行比较，所以只有给定量影响输出量。当出现外部扰动或内部扰动时，若没有人的干预，输出量将不能按照给定量所希望的状态去工作。

闭环控制系统是把输出量检测出来，经过物理量的转换，再反馈到输入端与给定量进行比较（相减），并利用比较后的偏差信号，经过控制器或调节器对控制对象进行控制，抑制内部或外部扰动对输出量的影响，从而减小输出量的误差。图 1.2 表示了闭环控制系统输入量、输出量和反馈量之间的关系。

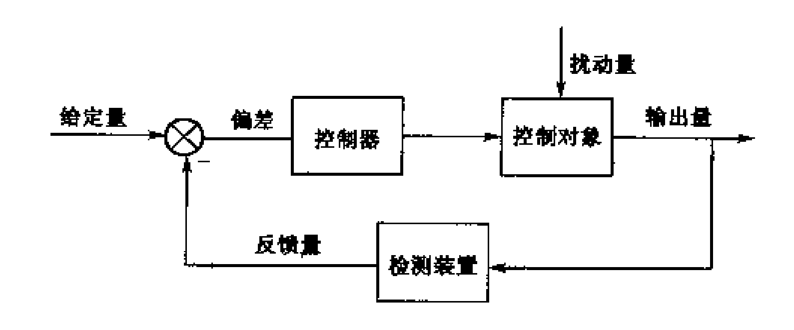


图 1.2　闭环控制示意图

这种系统把输出量直接或间接地反馈到输入端形成闭环，参与系统的控制，所以称为闭环控制系统。由于系统是根据负反馈原理按偏差进行控制的，因此也称为反馈系统或偏差控制系统。

在现代工业生产中，按照偏差控制的闭环系统种类繁多，尽管它们的控制任务不同，具体结构不完全相同，但是，检测偏差、利用偏差信号对控制对象进行控制，以减小或纠正输出量的偏差这一控制过程都是相同的。

这种系统的特点可归纳如下：

（1）在开环系统中，只有输入量对输出量产生控制作用；从控制结构上看，只有从输入端到输出端、从左向右的信号传递通道（该通道称为正向通道）。在闭环控制系统中，除正向通道外，还必须有从右向左、从输出端到输入端的信号传递通道，使输出信号也参与控制作用，该通道称为反馈通道。闭环控制系统就是由正向通道和负反馈通道组成的。

（2）为了检测偏差，必须直接或间接地检测出输出量，并将其变换为与输入量相同的物理量，以便与给定量相比较，得出偏差信号。所以闭环系统必须有检测环节、给定环节和比较环节。

（3）闭环控制系统是利用偏差量作为控制信号来纠正偏差的，因此系统中必须具有执行纠正偏差这一任务的执行机构。闭环系统正是靠放大了的偏差信号来推动执行机构，进一步对控制对象进行控制的。只要输出量与给定量之间存在偏差，就自动纠正输出量与期望值之间的误差，因此可以构成精确的控制系统。

反馈控制系统广泛地应用于各个工业部门，例如加热炉的温度控制、机器手的控制等。

在有些系统中，将开环控制和闭环控制结合在一起，构成一个开环—闭环控制系统，这种系统称为复合控制系统。

本书中提到的自动控制系统主要是指闭环控制系统。

1.2.2　闭环控制系统组成结构

闭环控制系统有各种不同的形式，但是概括起来，一般均由以下基本环节组成，如图 1.3 所示。

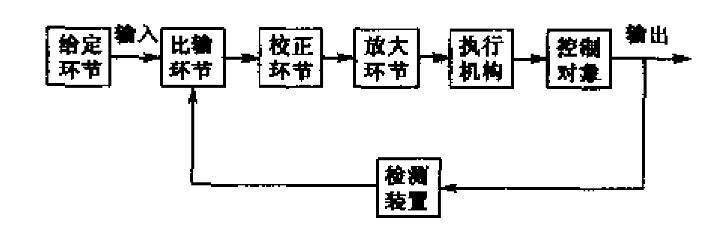


图 1.3　闭环控制系统结构图

（1）给定环节：它是设定被控制量的给定值的装置，如电位器等，给定环节的精度对被控制量的控制精度有被大的影响，现代的控制系统一般采用控制精度高的数字给定装置。

（2）比较环节：比较环节将所检测的被控制量和给定量进行比较，确定两者之间的偏差量。该偏差量由于功率较小或者物理性质不同，还不能直接作用于执行机构，所以在执行机构与比较环节之间还有中间环节。

（3）中间环节：中间环节一般是放大元件，将偏较信号变换成适于控制执行机构工作的信号。根据控制的要求，中间环节可以是一个简单的功率放大环节，或者是将偏差信号变被为适于执行结构工作的物理量，如较压伺服较大器。常常除了要求中间环节将偏差信号放大以外，还希望它能按某种规律对偏差信号进行运算，用运算的结构控制执行机构，以改善被控制量的稳态和暂态性能，这种中间环节常称为较正环节。

（4）执行机构：一般由传动较置和调节机构组成，执行机构直接作用于控制对象，被被控制量达到所变求的数值。

（5）控制对象或调节对象：它是指要进行控制的设备或过程，相应地，控制系统所控制的某个物理量就是系统的输出量或被控量，闭环控制系统的任务较是控制这些系统输出量的变化规律，以满足生产工艺的要求。

（6）较测装置或传感器：用于较测被控制量，并将其转换为与给定量统一的物理量。检测装置的精度和特性直被影响性制系统的较制品质，它是构成自动控制系统的关键元件，所以一般应要求被测装置的测量较度高、反应灵敏、性能稳定等。

在较制系统中，通常把比被环节、较正环节和放大环节合在一起称为控制装置。

1.2.3　反馈控制系馈品馈要求

在反馈控制系统中，当就动量或给定量（或给定量的变化规律）发生变化时，被控量偏离了给定量（或给定量的变化规律）而产生偏差，通过反馈控制的作用，经过短暂的过被过程，被控量又趋于或恢复到原来的稳态值，或按照就的给定量（或给定量的变化就律）稳定下来，这时系统从就来的平衡状态过渡到新的平衡状态。我们把被控量处于变化状态的过程称为动态或暂态，而把被控量处于相对稳定的状态称为静态或就态。

反馈较制系统品质要求可以归结为静定性（长期就定性）、快速性（相对稳定性）和准确性（精度）。

1. 稳定性

稳定性对于不同的系统有不同的要求。对于恒值系统，要求当系统受到扰动后，经过一定时间的调整能够回到原来的期望值；对于随动系统，要求被控制量始终跟踪参量的变化。

稳定性是对系统的基本要求，不稳定的系统不能实现预定任务。稳定性通常由系统的结构决定，与外界因素无关。

2. 快速性

快速性是指对过渡过程的形式和快慢提出要求，一般称为动态性能或暂态性能。一个自动控制系统还应能满足暂态性能的要求。如果控制对象的惯性很大，系统的反馈又不及时，则被控量在暂态过程中将产生过大的偏差，到达稳态的时间加长，并呈现各种不同的暂态过程。

一般说来，在合理的结构和适当的系统参数下，一个系统的暂态过程多属于衰减振荡过程，即被控量变化很快并产生超调，经过几个振荡后，达到新的稳定工作状态。为了满足生产工艺要求，往往要求系统的暂态过程不仅是稳定的，并且进行得越快越好，振荡程度越小越好。前者是暂态过程的稳定性问题，后者是暂态过程的性能问越。这些都是设计闭环控制系统时必须研究的问题。

3. 准确性

准确性通常用稳态误差来表示，所谓稳态误差是指系统达到稳态时，输出量的实际值和期望值之间的误差。这一性能表示稳态时的控制精度，一个设计合理的自动控制系统其稳态特性应能满足工艺的要求。

在参考输入信号作用下，当系统达到稳态后，其稳态输出与参考输入所要求的期望输出之差叫做给定稳态误差。显然，这种误差越小，表示系统的输出跟随参考输入的精度越高。

一个闭环控制系统往往在满足稳态精度和暂态品质之间存在着矛盾，例如要求稳态精度高，往往不能得到很好的暂态性能。因此必须兼顾这两方面的要求，根据具体情况合理地解决。

1.3 自动控制系统分类

自动控制系统广泛地应用于各类工业部门。随着生产规模的不断扩大和生产能力的不断提高，以及自动化技术和控制理论的发展，自动控制系统也日益复杂和日趋完善。例如，由单输入单输出的控制系统发展为多输入多输出的系统；由具有常规控制仪表和控制器的连续控制系统，发展到由计算机作为控制器的直接数字控制系统，从而实现最优控制。由于各式各样自动控制系统的不断发展，很难确切地对自动控制系统进行分类。现将常见的几种自动控制系统概括介绍如下。

1.3.1　线性系统和非线性系线

按不同系统的特征方程式，可将自动控制系统分为线性系统和非线性系统。

线性控制系统是由线性元件组成的系统，该系统的特征方程式可以用线性微分方程描述。叠加性和齐次性是鉴别系统是否为线性系统的根据。线性微分方程的各项系数为常数时，称为线性定常系统。线性定常系统可以用拉普拉斯变换解微分方程，并由此定义出系统传递函数这一系统动态数学模型。根轨迹法和频率法就是在这一基础上发展起来的分析和设计线性系统的有效方法。多输入多输出系统所采用的状态空间、传递矩阵等分析方法，将在有关章节中论述。

如果系统微分方程的系数与自变量有关，则为非线性微分方程，由非线性微分方程描述的系统称为非线性控制系统。在自动控制系统中，即使只含一个非线性环节，这一系统也是非线性的。

对于非线性控制系统的理论与研究远不如线性控制系统那样完整，一般只能满足于近似的定性描述和数值计算。任何物理系统的特性，精确地说都是非线性的，但在误差允许范围内，可以将非线性特性线性化，近似地用线性微分方程来描述，这样就可以按照线性系统来处理。

非线性系统的暂态特性与其初始条件有关，从这一点来看，它与线性系统有很大的区别。例如当偏差的初始值很小时，系统的暂态过程是稳定的，而当偏差量的初值较大时，则可能是不稳定的。线性系统的暂态过程与初始条件无关。

1.3.2　离散系统和连续系统

从数学模型角度而言，连续系统各部分信号均以模拟的连续函数形式表示，以前的大部分闭环系统都属于连续系统。

从数学模型角度而言，离散系统的某一处或几处信号是以脉冲序列或数字形式表示的。目前的计算机控制系统都属于离散系统。

离散系统的主要特点是：在系统中使用脉冲开关或采样开关，将连续信号转变为离散信号。通常对以离散信号取脉冲形式的系统，称为脉冲控制系统；而对于采样数字计算机或数字控制器，其离散信号以数字形式传递称为采样数字控制系统。图 1.4 是典型的采样数字控制系统结构示意图。

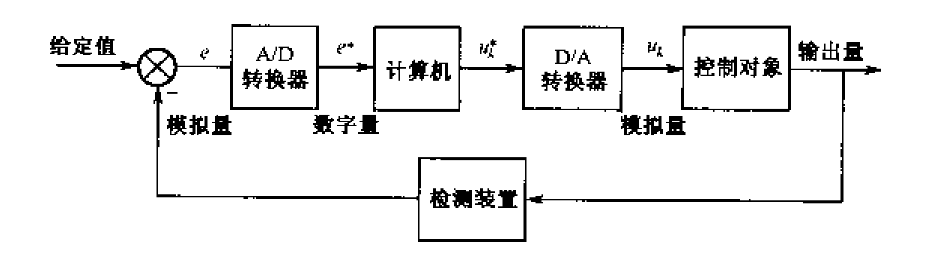


图 1.4　采样数字控制系统结构示意图

由于被控对象的输入量和输出量是模拟信号，而计算机的输入量和输出量是数字信号，所以要有将模拟量转换为数字量的模数转换装置（A/D）和把数字量转换为模拟量的数模转换装置（D/A）。

研究离散系统的方法和研究连续系统的方法类似。

1.3.3　恒值系统和统动系统

在现代生产应用中使用最多的闭环自动控制系统，按给定量的不同特征，可将系统分为恒值系统和随动系统。

恒值系统往往要求被控制量保持在恒定值，其给定量是不变的，如恒温、恒速、恒压等自动控制系统。

在随动系统中，给定量是按照事先不知道的时间函数变化，要求输出量跟随给定量的变化而变化，因此也称为同步随动系统，如自动火炮的控制系统。

1.4　自动控制系统仿真基本概念

系统仿真作为一种特殊的试验技术，在 20 世纪 30～90 年代的半个多世纪中经历了飞速发展，到今天已经发展成为一种真正的、系统的实验科学。伴随着第一台电子管电子计算机的诞生和以相似理论为基础的模拟技术的应用，仿真作为一种研究、发展新产品、新技术的科学手段，在航变、航天、造船、兵器等与国防科研相关的行业中首先发展起来，并显示了巨大的社会效益和经济效益。

以武器的作战使用训练为例，1930 年左右，变国陆、海军航空队就使用了林克式仪表飞行模拟训练器。当时其经济效益相当于每年节约 1.3 亿美元，而且少牺牲 524 名飞行员，此后，固定基座及三自由度飞行模拟座舱陆续大量投入使用。1950～1953 年，美国首先利用计算机来模拟战争，防变兵力或地空作战被认为是具有最大训练潜力的应用范畴。20 世纪 60 年代，目标探测、模获、跟踪和电子对抗已经进入了仿真系统。70 年代利用放电影方式，在大球幕内实现了多目标、飞机一导弹作战演习。随着 80 年代数字计算机的离速发展，训练仿真开始蓬勃发展，甚至呈现了两个新概念，即武器系统研制与训练装置的开发同步进行以及训练装置作为武器系统可嵌入的组成部分而进入整个计算机软件系统。至于武器的控制与制导（C&G）系统研制、试验与定型中仿真技术的应用则更为普遍。在 80 年代对于导弹的研制中，由于采用仿真使飞行试验数量减小了 30%～40%，节约研制经费 10%～40%，缩短周模 30%～60%，这足以说明系统仿真在工程应用中的重大意义。

仿真的基本思想是利用物理的或数学的模型来类比模仿现实过程，以寻求对真实过程的认识，它所遵循的基本原则是相似性原理。

1.4.1　计算机仿真基本概念

计算机仿真是基于所建立的系统仿真模型，利用计算机对系统进行分析与研究的技术与方法。

1.4.1.1　模型

模型是对现实系统有关结构信息和行为的某种形式的描述，是对系统特征与变化规律的一种定量抽象，是人们认识事物的一种手段或工具。

模型可以分为以下三类：

（1）物理模型，指不以人的意志为转移的客观存在的实体。例如，飞行器研制中的飞行模型、船舶制造中的船舶模型等。

（2）数学模型，指从一定的功能或结构上进行相似，用数学的方法来再现原型的功能或结构特征。

（3）仿真模型，指根据系统的数学模型，用仿真语言转化为计算机可以实现的模型。

1.4.1.2　仿真分类

可以从模型角度和计算机类型角度对不同的仿真系统进行分类。

1. 按模型分类

（1）物理仿真：采用物理模型，有实物介入，具有效果逼真、精度高等优点，但造价高或耗时长，大多在一些特殊场合下采用（如导弹、卫星一类飞行器的动态仿真，发电站综合调度仿真与培训系统等），具有实时、在线的特点。

（2）数学仿真：采用数学模型。它在计算机上进行，具有非实时、离线的特点，经济、快速、实用。

2. 按计算机类型分类

（1）模拟仿真：采用数学模型，在模拟计算机上进行的仿真实验。这是一种早期的仿真手段，现在基本被淘汰。它的特点是描述连续物理系统的动态过程比较自然、逼真，具有仿真速度快、失真小、结果可靠的优点，但受元器件性能影响，仿真精度较低，对计算机控制系统的仿真较困难，自动化程度低。模拟计算机的核心是运算部分，它由我们熟知的“模拟运算放大器”为主要构成部件。

（2）数字仿真：采用数学模型，在数字计算机上借助数值计算方法所进行的仿真实验。它是在 20 世纪 60 年代随着计算机的发展而发展起来的，其特点是计算与仿真的精度较高。理论上计算机的字长可以根据精度要求来“随意”设计，因此其仿真精度可以是无限的，但是由于受到误差积累、仿真时间等因素影响，其精度也不易定得太高。它对计算机控制系统的仿真比较方便，仿真实验的自动化程度较高，可方便地实现显示、打印等功能，但计算速度比较低，在一定程度上影响到仿真结果的可信度。随着计算机技术的发展，速度问题会在不同程度上有所改进与提高。数字仿真没有专用的仿真软件支持，需要设计人员用高级程序语言编写求解系统模型及结果输出程序。

（3）混合仿真：结合了模拟仿真与数字仿真的技术与特点。

（4）现代计算机仿真：采用先进的微型计算机，基于专用的仿真软件、仿真语言来实现，其数值计算功能强大，易学易用。它是在 20 世纪 80 年代发展起来的，是当前主流的仿真技术与方法。

1.4.1.3　仿真应用

仿真技术有着广泛的应用，而且应用的深度和广度也越来越大，目前主要应用在以下方面：

（1）航空与航天工业，包括飞行器设计中的三级仿真体系（即纯数学仿真）、半实物仿真、实物仿真或模拟飞行实验，飞行员及宇航员训练用飞行仿真模拟器等。

（2）电力工业，包括电力系统动态模型实验，电力系统负荷分配、瞬态稳定性以及最优潮流控制，电站操作人员培训模拟系统等。

（3）原子能工业，包括模拟核反应堆，核电站仿真器，用来训练操作人员以及研究异常故障的排除处理等。

（4）石油、化工及冶金工业。

（5）非工程领域，如医学、社会学、宏观经济和商业策略的研究等。

1.4.1.4　仿真技术应用意义

仿真技术的应用具有重要的意义，主要体现在以下方面：

（1）经济。大型、复杂系统直接实验是十分昂贵的，如空间飞行器一次飞行实验的成本约在 1 亿美元左右，而采用仿真实验仅需其成本的 1/10～1/5，而且设备可以重复使用。

（2）安全。某些系统，如载人飞行器、核电装置等，直接实验往往会有很大的危险，甚至是不允许的，而采用仿真实验可以有效降低危险程度，对系统的研究起到保障作用。

（3）快捷。提高设计效率，如电路设计、服装设计等。

（4）具有优化设计和预测的特殊功能。对一些真实系统进行结构和参数的优化设计是非常困难的，这时仿真可以发挥它特殊的优化设计功能。再如在非工程系统，如社会、管理、经济等系统，由于其规模及复杂程度巨大，直接实验几乎是不可能的，这时通过仿真技术与方法的应用可以获得对系统的某种超前认识。

1.4.2　自动控制系统仿真

自动控制系统仿真是系统仿真的一个重要分支，它是一门涉及自动控制理论、计算数学、计算机技术、系统辨识、控制工程以及系统科学的综合性新型学科。它为控制系统的分析、计算、研究、综合设计以及自动控制系统的计算机辅助教学等提供了快速、经济、科学及有效的手段。

自动控制系统仿真就是以自动控制系统模型为基础，采用数学模型替代实际控制系统，以计算机为工具，对自动控制系统进行实验、分析、评估及预测研究的一种技术与方法。

1.4.3　自动控制系统计算机仿真基本过程

自动控制系统仿真包括以下几个基本步骤：问题描述、模型建立、仿真实验、结果

分析，其流程如图 1.5 所示。

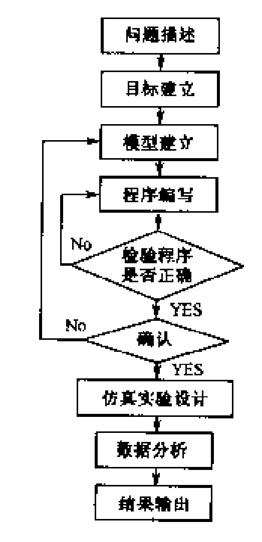


图 1.5　计算机仿真流程图

1. 建立数学模型

控制系统模型，是指描述控制系统输入、输出变量以及内部各变量之间关系的数学表达式。控制系统模型可分为静态模型和动态模型，静态模型描述的是自动控制系统变量之间的静态关系，动态模型描述的是自动控制系统变量之间的动态关系。最常用、基本的数学模型是微分方程与差分方程。控制系统数学模型的建立将在第 4 章中做详细的论述。

2. 建立仿真模型

由于计算机数值计算方法的限制，有些数学模型是不能直接用于数值计算的，如微分方程，因此原始的数学模型必须转换为能够进行系统仿真的仿真模型。例如，在进行连续系统仿真时，就需要将微分方程建样的数学模型通过拉普拉斯变换转换成传递函数结构的仿真模型。

3. 编写仿真程序

控制系统的仿真涉及很多相关联的量，这些量之间的联系要通过编制程序来实现，常用的数值仿真编程语言有 C、FORTRAN 等，近年来发展迅速的综合计算仿真软件，如 MATLAB 也可以用来编写仿真模序，而且模写起来非常迅速、界面友好，已得到广泛应用，本书就将结合 MATLAB 来对自动控制系统仿真进行阐述。

4. 进行仿真实验并分析实验结果

在完成以上工作后，就可以进行仿真实验了，通过对仿真结果的分析来对仿真模型与仿真程序进行检验和修改，如此反复，直至达到满意的实验效果为止。

1.4.4　计算机仿真技术发展趋势

随着计算机技术的发展与进步，与之紧密结合的计算机仿真技术也得到了飞速发展，其发展趋势主要体现在以下方面。

(1) 硬件方面：基于多 CPU 并行处理技术的全数字仿真将有效提高仿真系统的速度，大大增强数字仿真的实时性。

(2) 应用软件方面：直接面向用户的数字仿真软件不断推陈出新，各种专家系统与智能化技术将更深入地应用于仿真软件开发之中，使得在人机界面、结果输出、综合评判等方面达到更理想的模界。

(3) 分布式数字仿真：充分利用网络技术进行分布式仿真，投资少，效果好。

(4) 虚拟现实技术：综合了计算机图形技术、多媒体技术、传感器技术、显示技术以及仿真技术等多学科，使人仿佛置身于真实环模之中，这就是“仿真”追求的最终目标。

1.5 MATLAB 与控制系统仿真

在控制系统仿真初期，往往需要仿真技术人员自己用 BASIC 等语言去编写数值计算程序。例如，如果想求得系统的阶跃相应数据并绘制阶跃响应曲线，则首先需要编写一个求解微分方程的子程序，然后将原系统模型输入给计算机，通过计算机求出阶跃响应数据，然后再编写一个画图的子程序，将所得的数据以曲线的方式绘制出来。显然，求解这样简单的问题需要花费很多的时间，并且由于没有纳入规范，往往不能保证求解结果的正确性。

自 MATLAB 面世以来，其应用范围越来越广，软件工具越来越完善，特别是 MATLAB 的控制系统工具箱及 Simulink 的问世，给控制系统的分析和设计带来了极大的方便，现已成为风行国际的、有力的控制系统计算机辅助分析工具。

MATLAB 具有以下主要特点，非常适合于控制系统的仿真：

（1）强大的运算功能。MATLAB 提供了向量、数组、矩阵、复数运算，求解高次微分方程、常微分方程的数值积分等强大的运算功能，这些运算功能使控制理论及控制系统中经常遇到的计算问题得以顺利解决。

（2）特殊功能的 TOOLBOX 工具箱。MATLAB 的 TOOLBOX 工具箱包括控制领域里常用的算法包，如模糊控制工具箱、鲁棒控制工具箱等，这些工具箱使得控制系统的计算与仿真变得方便。

（3）高效的编程效率。MATLAB 提供了丰富的库函数，这些库函数都可以直接调用，而不必将其子程序的命令或语句逐一列出，大大提高了编程效率。在科学与工程应用的数值计算领域里，与传统使用 BASIC、FORTRAN 和 C 等语言设计程序相比，编程效率将提高好几倍。

（4）简单易学的编程语言。MATLAB 的编程语言是脚本语言，这种解释性的语言简单易学。

（5）方便友好的编程环境。MATLAB 接供了友好的用户界面和方便的帮助系统，十分方便操作者使用。

随着 MATLAB 软件的不断升接以及功能强大的 TOOLBOX 的出现，MATLAB 将成为自动控制系统计算与仿真一个越来越强有力的工具，使控制系统的计算与仿真箱传统方法发生革命性的变化，MATLAB 正成为国内外控制领域内最流行的计算与仿真软件。

第 2 章　MATLAB 基础知识

2.1 引　　言

经过 20 余年的补充与完善以及多个版本的升级换代，MATLAB 已发展至 7.0 版本。MATLAB 是一个包含众多工程计算、仿真功能及工具的庞大系统，是目前世界上最流行的仿真计算软件。MATLAB 软件和工具箱（TOOLBOX）以及 Simulink 仿真工具，为自动控制系统的计算与仿真提供了强有力的支持。

本章介绍 MATLAB 的产生与发展过程，对该软件最新版本 MATLAB 7.0 的常用工具箱及最新特点进行介绍，而后对 MATLAB 的基础知识和基本命令进行比较详细的描述。通过本章，读者对 MATLAB 能有一个比较全面的了解，并能熟练使用 MATLAB 的常用功能。

2.1.1 MATLAB 发展历程

MATLAB 的产生是与数学计算紧密联系在一起的。1980 年，美国新墨西哥州大学计算机系主任 Cleve Moler 在给学生讲授线性代数课程时，发现学生在高级语言编程上花费很多时间，于是着手编写供学生使用的 Fortran 子程序库接口程序，他将这个接口程序取名为 MATLAB（即 Matrix Laboratory 的前三个字母的组合，意为“矩阵实验室”）。这个程序获得了很大的成功，受到学生的广泛欢迎。

20 世纪 80 年代初期，Moler 等一批数学家与软件专家组建了 MathWorks 软件开发公司，继续从事 MATLAB 的研究和开发，1984 年推出了第一个 MATLAB 商业版本，其核心是用 C 语言编写的。而后，它又添加了丰富多彩的图形图像处理、多媒体、符号运算以及与其他流行软件的接口功能，使得 MATLAB 的功能越来越强大。

MathWorks 公司正式推出 MATLAB 后，于 1992 年推出了具有划时代意义的 MATLAB 4.0 版本；1999 年推出的 MATLAB 5.3 版在很多方面进一步改进了 MATLAB 的功能，随之推出的全新版本 Simulink 3.0 也达到了很高的档次；2000 年 10 月推出的 MATLAB 6.0 版本，在操作界面上有了很大的改观，同时还给出了程序发布窗口、历史信息窗口和变量管理窗口等，为用户提供了极大的方便；2001 年 6 月，MATLAB 6.1 版即 Simulink 4.0 版问世，功能已经十分强大，其虚拟显示工具箱更给仿真结果三维视景下显示带来了新的解决方案；2001 年 6 月推出了 MATLAB Release 13，即 MATLAB 6.5/Simulink 5.0，在核心数值算法、界面设计、外部接口、应用桌面等诸多方面有了极大的改进；2004 年 9 月正式推出 MATLAB Release 14，即 MATLAB 7.0/Simulink 6.0，其功能在原有的基础上又有了进一步的改进，它是 MATLAB 目前最新的版本。

MATLAB 经过几十年的研究与不断完善，现已成为国际上最为流行的科学计算与工

程计算软件工具之一，现在的 MATLAB 已经不仅仅是一个最初的“矩阵实验室”了，它已发展成为一种具有广泛应用前景、全新的计算机高级编程语言，可以说它是“第四代”计算机语言。自 20 世纪 90 年代，在美国和欧洲大学中，将 MATLAB 正式列入研究生和本科生的教学计划，MATLAB 软件已成为应用代数、自动控制理论、数理统计、数字信号处理、时间序列分析、动态系统仿真等课程的基本教学工具，成为学生所必须掌握的基本软件之一。在研究单位和工业界，MATLAB 也成为工程师们必须掌握的一种工具，被认做进行高效研究与开发的首选软件工具。

2.1.2 MATLAB 系统构成

MATLAB 系统由 MATLAB 开发环境、MATLAB 数学函数库、MATLAB 语言、MATLAB 图形处理系统和 MATLAB 应用程序接口（API）五大部分构成。

1. MATLAB 开发环境

MATLAB 开发环境是一套方便用户使用 MATLAB 函数和文件的工具集，其中许多工具是图形化用户接口。它是一个集成化的工作空间，可以让用户输入、输出数据，并提供了 M 文件的集成编译和调试环境。它包括 MATLAB 桌面、命令窗口、M 文件编辑调试器、MATLAB 工作空间和在线帮助文档。

2. MATLAB 数学函数库

MATLAB 数学函数库包括了大量的计算算法，从基本运算（如加法、正弦等）到复杂算法，如矩阵求逆、贝塞尔函数、快速傅里叶变换等。

3. MATLAB 语言

MATLAB 语言是一个高级的基于矩阵/数组的语言，它有程序流控制、函数、数据结构、输入/输出和面向对象编程等特色。用户既可以用它来快速编写简单的程序，也可以用来编写庞大复杂的应用程序。

4. MATLAB 图形处理系统

图形处理系统使得 MATLAB 能方便地图形化显示向量和矩阵，而且能对图形添加标注和打印。它包括强力的二维、三维图形函数、图像处理和动画显示等函数。

5. MATLAB 应用程序接口（API）

MATLAB 应用程序接口（API）是一个使 MATLAB 语言能与 C、Fortran 等其他高级编程语言进行交互的函数库，该函数库的函数通过调用动态链接库（DLL）实现与 MATLAB 文件的数据交换，其主要功能包括在 MATLAB 中调用 C 和 Fortran 程序，以及在 MATLAB 与其他应用程序间建立客户/服务器关系。

2.1.3 MATLAB 7.0 工具箱

MATLAB 拥有一个专用的家族产品，用于解决不同领域的问题，称之为工具箱

（TOOLBOX），工具箱用于 MATLAB 的计算和画图，通常是 M 文件和高级 MATLAB 语言集合，以使用户可以方便地修改函数和源代码，或增加新的函数。用户还可以很方便地结合使用不同的工具箱中的技术来设计针对某个问题的用户解决方案。MATLAB 每年都会增加一些新的工具箱，所以，在一般情况下，工具箱的列表不是固定不变的，有关 MATLAB 工具箱的最新信息可以在 http://www.mathworks.com/products 中看到。

较为常见的 MATLAB 工具箱包括以下几类。

1. 控制类工具箱

- 控制系统工具箱（Control Systems Toolbox）
- 系统辨识工具箱（System Identification Toolbox）
- 鲁棒控制工具箱（Robust Control Toolbox）
- 模糊逻辑工具箱（Fuzzy Logic Toolbox）
- 神经网络工具箱（Neural Network Toolbox）
- 频域系统辨识工具箱（Frequency Domain System Identification Toolbox）
- 模型预测控制工具箱（Model Predictive Control Toolbox）
- 多变量频率设计工具箱（Multivariable Frequeucy Design Toolbox）

2. 应用数学类工具箱

- 最优工具箱（Optimization Toolbox）
- 样条工具箱（Spline Toolbox）
- 统计工具箱（Statistics Toolbox）
- 偏微分方程工具箱（Partial Differential Equation Toolbox）

3. 信号处理类工具箱

- 信号处理工具箱（Signal Processing Toolbox）
- 滤波器设计工具箱（Filter Design Toolbox）
- 通信工具箱（Commnnication Toolbox）
- 小波分析工具箱（Wavelet Toolbox）
- 高阶谱分析工具箱（Higher Order Spectral Analysis Toolbox）

4. 其他常用的工具箱

- 符号数学工具箱（Symbolic Math Toolbox）
- 虚拟现实工具箱（Virtual Reality Toolbox）

MATLAB 7.0 对以前的版本进行了升级，主要新增了以下一些功能强大的工具箱。

- 定点工具箱（Fixed-Point Toolbox）
- 遗传算法和直接搜寻工具箱（Genetic Algorithm and Direct Search Toolbox）
- 射频工具箱（RF Toolbox）
- 生物信息工具箱（Bioinformatics Toolbox）
- OPC 服务器工具箱（OPC Toolbox）

2.1.4　MATLAB 7.0/Simulink 6.0 最新特点

MATLAB 既是高级科学计算语言，又是进行数据分析和算法开发的集成开发环境。MATLAB 7.0 在 MATLAB 6.5 的基础上进行了很多改善和提高；Simulink 6.0 在 Simulink 5.0 的基础上也增加了不少新功能，使得这一软件的性能大大提高。

2.1.4.1　MATLAB 7.0 最新特点

MATLAB 7.0 在编程环境、代码效率、数据可视化、数学计算、文件 I/O 等方面进行了升级，包括以下几方面。

1. *开发环境方面*

（1）重新设计的桌面环境，针对多文档界面应用提供了简便的管理和访问方法，允许用户自定义桌面外观、创建常用命令的快捷方式；

（2）增强数组编辑器（Array Editor）和工作空间浏览器（Workspace Brower）功能，用于数据的显示、编辑和处理；

（3）在当前目录浏览器　（Current Directory Browser）工具中，增加代码效率分析、覆盖度分析等功能；

（4）M-Lint 编码分析，辅助用户完成程序性能分析，提高程序执行效率；

（5）增强 M 文件编辑器(M-Editor)，支持多种格式源代码文件可视化编辑，如 C/C++、HTML、Java 等。

2. *编程方面*

（1）文件创建嵌套函数（Nested Function），提供更灵活的代码模块化方式；

（2）匿名函数（Anonymous Function）功能，支持在命令行或者脚本文件中创建单行函数（Single Line Fuuction）；

（3）支持条件分支断点，可以在条件分支语句中进行程序中断调试；

（4）模块化注释，支持为代码段注释。

3. *数学运算方面*

（1）支件整数算术运算；

（2）支持单精度数据类型运算，包括基本算术运算、线性代数、FFT 等；

（3）使用更强大的计算算法包 Qhull 2002.1，提供更丰富的算法文件；

（4）linsolve 函数用于进行线性代数方程求解；

（5）ODE 求解器能够处理隐性微分方程组以及多点边界问题。

4. *图形和 3-D 可视化方面*

（1）新图形窗体界面；

（2）直接从图形窗件生成 M 代码，可以完成用户自定义绘图；

（3）增强图形窗体注释；

（4）数据侦测工具（Data Exploration Tools），提供丰富的数据观测手段；
（5）自定义图形对象，提供丰富的图形显示能力；
（6）GUIDE 新增对用户界面面板和 ActiveX 控件支持；
（7）增强句柄图形对象支持完整的 TeX 和 LaTeX 字符集。

5. 文件 I/O 和外部接口方面

（1）新增文件 I/O 函数，支持读取任意格式文本数据文件，并且支持写入 Excel 和 HDF5 格式数据文件；
（2）具有压缩功能的 MAT 文件格式，支持快速数据文件 I/O 能力；
（3）javaaddpath 函数，无须重新启动 MATLAB 即可完成 Java 类的加载、删除等功能；
（4）支持 COM、服务器事件以及 VBS；
（5）支持 SOAP，使用网络服务；
（6）支持 FTP 对象，直接访问 FTP 服务器；
（7）支持 Unicode 编码格式，增强 MAT 文件字符集。

6. 性能与系统平台支持方面

（1）JIT 加速器支持所有数值数据类型；
（2）Windows XP 系统下支持 3GB 内存访问。

2.1.4.2　Simulink 6.0 最新特点

Simulink 是一个交互式动态系统建模、仿真和分析图形环境，是一个进行基于模型的嵌入式系统开发的基础开发环境。Simulink 可以针对控制系统、信号处理和通信系统等进行系统建模、仿真、分析等工作。Simulink 6.0 改善了性能，并且针对大规模系统开发进行了性能优化，主要包括以下方面。

1. 大系统建模方面

（1）支持将大系统模型分割为不同的文件，每个文件为独立的系统模型；
（2）支持系统不同模型文件独立仿真测试；
（3）增强系统集成手段，支持配置管理和版本控制软件；
（4）递增式模型加载与代码生成功能；
（5）针对大模型系统提高运行速度和效率；
（6）模型工作空间（Model Workspace），每个模型都用于独立的工作空间进行参数管理和数据处理；
（7）增强总线支持。

2. Simulink 和 Stateflow 方面

（1）统一的模型浏览器（Model Explorer），用于浏览、维护、配置、搜索、定义所有模型中相关的参数、数据对象和属性；
（2）统一的仿真和代码生成选项；

（3）支持创建并保存多种仿真和代码生成选项；
（4）数据管理和可视化；
（5）新增数据对象属性；
（6）可选数据记录增加测试点，无须在模型中增加额外的模块；
（7）I/O 管理，可以将必要的信号源和信宿连接到模型而不需要增加模块。

3. MATLAB 支持方面

（1）使用嵌入式 MATLAB（Embedded MATLAB）功能引入算法并支持 C 代码生成；
（2）增强 M 语言 S 函数的支持。

2.2　MATLAB 桌面操作环境

MATLAB 为用户提供了全新的桌面操作环境，了解并熟悉这些桌面操作环境是使用 MATLAB 的基础。下面介绍 MATLAB 的启动、主要功能菜单、命令窗口、工作空间、文件管理和帮助管理等。

2.2.1　MATLAB 启动和退出

以 Windows 操作系统为例，进入 Windows 后，选择“开始”→“程序”→“Matlab 7.0”，便可以进入如图 2.1 所示的 MATLAB 主窗口；如果安装时选择在桌面上生成快捷方式，也可以单击快捷方式直接启动。



图 2.1　MATLAB 主窗口

在启动 MATLAB、命令编辑区显示帮助信息后，将显示符号“|”，符号“|”表示

MATLAB 已准备好，正等待用户输入命令，这时就可以在提示符“I”后面键入命令，按下回车键后，MATLAB 就会解释执行所输入的命令，并在命令后面给出计算结果。如果在输入命令后再以分号结束，则不会显示结果。

退出 MATLAB 系统的方式有两种：

（1）在文件菜单（File）中选择“Exit”或“Quit”。

（2）用鼠标单击窗口右上角的关闭图标。

2.2.2　MATLAB 主菜单及功能

打开 MATLAB 主窗口后，即弹出其主菜单栏，主菜单栏各菜单项及其下拉菜单的功能如下所述。

1. File 主菜单项

单击 File 主菜单项或同时按下“Alt+F”组合键，弹出如图 2.2 所示的 File 下拉菜单。其中，带下划线的字母表示快捷键，即单击该字母键也可执行相应的功能。

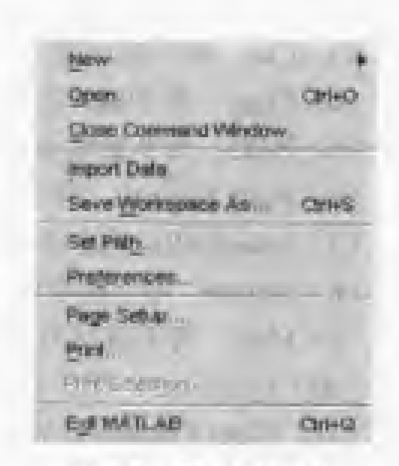


图 2.2　File 下拉菜单

（1）New：用于建立新的.m 文件、图形、模型和图形用户界面；

（2）Open：用于打开 MATLAB 的.m 文件、.fig 文件、.mat 文件、.mdl 文件、.cdr 文件等，也可通过快捷键“Ctrl+O”来实现此项操作；

（3）Close Command Window：关闭命令窗口；

（4）Import Data：用于从其他文件导入数据，单击后弹出对话框，选择导入文件的路径和位置；

（5）Save Workspace As：用于把工作空间的数据存放到相应的路径文件中；

（6）Set Path：设置工作路径；

（7）Preferences：用于设置命令窗的属性，单击该选项弹出如图 2.3 所示属性画面；

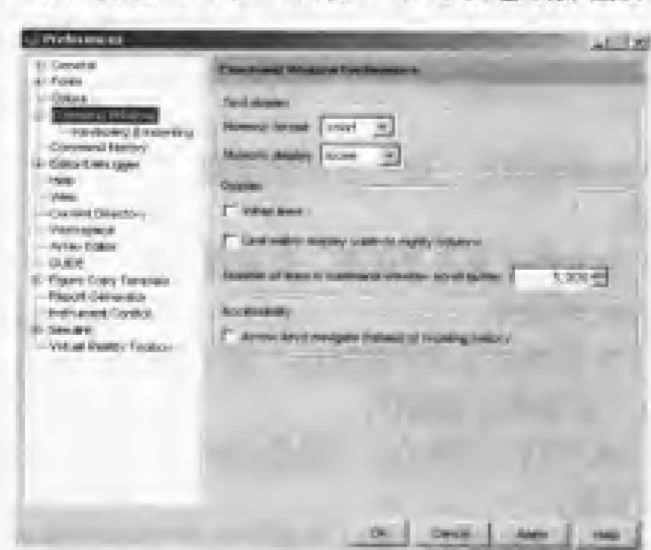


图 2.3　命令窗口属性对话框

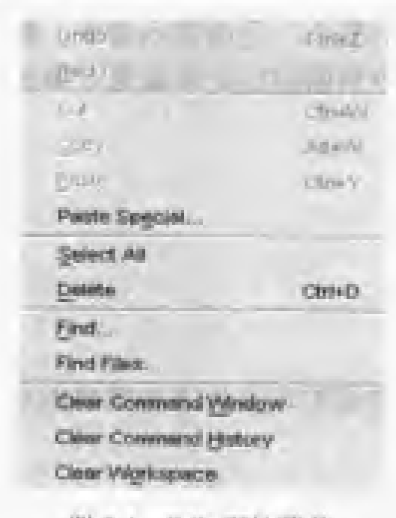


图 2.4　Edit 下拉菜单

（8）Page Setup：用于页面设置；

（9）Print：用于设置打印属性；

（10）Print Selection：用于对选择的文件数据进行打印设置；

（11）Exit MATLAB：退出 MATLAB 桌面操作环境。

2．Edit 主菜单项

单击 Edit 主菜单项或同时按下“Alt+E”组合键，弹出如图 2.4 所示的下拉菜单。

（1）Undo：用于撤销上一步操作，也可通过快捷键“Ctrl+Z”来实现此项操作；

（2）Redo：用于重新执行上一步操作；

（3）Cut：用于剪切选中的对象，也可通过快捷键“Ctrl+W”来实现此项操作；

（4）Copy：用于复制选中的对象，也可通过快捷键“Alt+W”来实现此项操作；

（5）Paste：用于粘贴剪贴板上的内容，也可通过快捷键“Ctrl+Y”来实现此项操作；

（6）Paste Special：用于特定内容的粘贴；

（7）Select All：用于全部选择；

（8）Delete：用于删除所选的对象，也可通过快捷键“Ctrl+D”来实现此项操作；

（9）Find：用于查找所需选择的对象；

（10）Find Files：用于查找所需文件；

（11）Clear Command Window：用于清除命令窗口区的对象；

（12）Clear Command History：用于清除命令窗口区的历史记录；

（13）Clear Workspace：用于清除工作区的对象。

3．Debug 主菜单项

单击 Debug 主菜单项或同时按下“Alt+B”组合键，弹出如图 2.5 所示的下拉菜单。

（1）Open M-Files when Debugging：用于调试时打开 M 文件；

（2）Step：用于单步调试程序，也可通过快捷键“F10”来实现此项操作；

（3）Step in：用于单步调试进入子函数，也可通过快捷键“F11”来实现此项操作；

（4）Step Out：用于单步调试从子函数跳出，也可通过快捷键“Shift+F11”来实现此项操作；

（5）Continue：程序执行到下一断点，也可通过快捷键“F5”来实现此项操作；

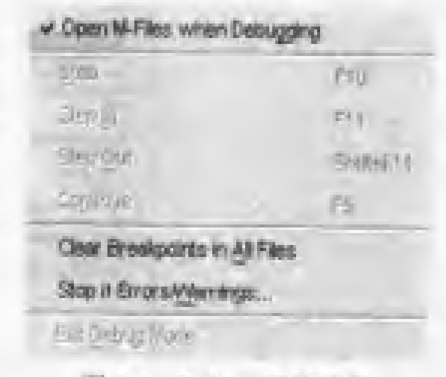


图 2.5　Debug 下拉菜单

（6）Clear Breakpoints in All Files：清除所有打开文件中的断点；

（7）Stop if Errors/Warnings：在程序出错或报警处停止往下执行；

（8）Exit Debug Mode：退出调试模式。

4. Desktop 主菜单项

单击 Desktop 主菜单项或同时按下“Alt+D”组合键，弹出如图 2.6 所示的下拉菜单。

（1）Undock Command Window：将命令窗口变为全屏显示，并设为当前活动窗口；

（2）Desktop Layout：单击该项后，弹出如图 2.7 所示的子菜单；用于工作区的设置，其设置选项包括系统默认设置项（Default）、单独命令窗口项（Command Window Only）、命令历史窗口和命令窗口项（History and Command Window）、全部标签项显示（All Tabbed）；

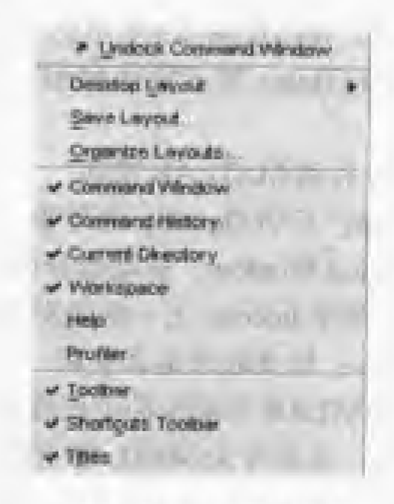


图 2.6 Desktop 下拉菜单

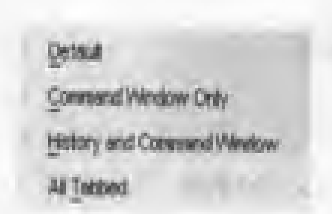


图 2.7 Desktop Layout 弹出子菜单

（3）Save Layout：保存选定的工作区设置；

（4）Organize Layouts：管理保存的工作区设置；

（5）Command Window：命令窗口项，选择该项，屏幕上便会显示相应窗口；

（6）Command History：命令历史窗口项，选择该项，屏幕上便会显示相应窗口；

（7）Current Directory：当前路径窗口项，选择该项，屏幕上便会显示相应窗口；

（8）Workspace：工作窗口项，选择该项，屏幕上便会显示相应窗口；

（9）Help：帮助窗口项，选择该项，屏幕上便会显示相应窗口；

（10）Profiler：轮廓图窗口项，选择该项，屏幕上便会显示相应窗口；

（11）Toolbar：显示或隐藏工具栏选项；

（12）Shortcuts Toolbar：显示或隐藏快捷方式选项；

（13）Titles：显示或隐藏标题栏选项。

5. Window 主菜单项

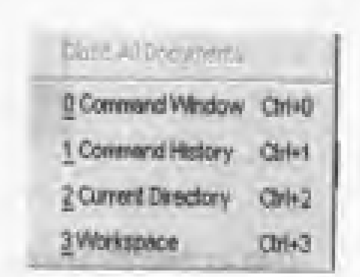


图 2.8 Window 下拉菜单

单击 Window 主菜单项或同时按下“Alt+W”组合键，弹出如图 2.8 所示的下拉菜单。

（1）Close All Documents：关闭所有文档；

（2）0 Command Window：选定命令窗口为当前活动窗

口，也可通过快捷键“Ctrl+0”来实现此项操作；

（3）1 Command History：选定命令历史窗口为当前活动窗口，也可通过快捷键“Ctrl+1”来实现此项操作；

（4）2 Current Directory：选定当前路径窗口为当前活动窗口，也可通过快捷键“Ctrl+2”来实现此项操作；

（5）3 Workspace：选定工作空间窗口为当前活动窗口，也可通过快捷键“Ctrl+3”来实现此项操作。

6．Help 主菜单项

单击 Help 主菜单项或同时按下“Alt+H”组合键，弹出如图 2.9 所示的下拉菜单。



图 2.9　Help 下拉菜单

（1）Full Product Family Help：显示所有 MATLAB 产品的帮助信息；

（2）MATLAB Help：启动 MATLAB 帮助；

（3）Using the Desktop：启动 Desktop 的帮助；

（4）Using the Command Window：启动命令窗口的帮助；

（5）Web Resources：显示 Internet 上一些相关的资源网址；

（6）Check for Updates：检查软件是否更新；

（7）Demos：调用 MATLAB 所提供的范例程序；

（8）About MATLAB：显示有关 MATLAB 的信息。

2.2.3　MATLAB 命令窗口

MATLAB 的命令窗口，如图 2.10 所示，它用于 MATLAB 命令的交互操作，它具有两大主要功能：

（1）提供用户输入命令的操作平台，用户通过该窗口输入命令和数据；

（2）提供命令执行结果的显示平台，该窗口显示命令执行的结果。

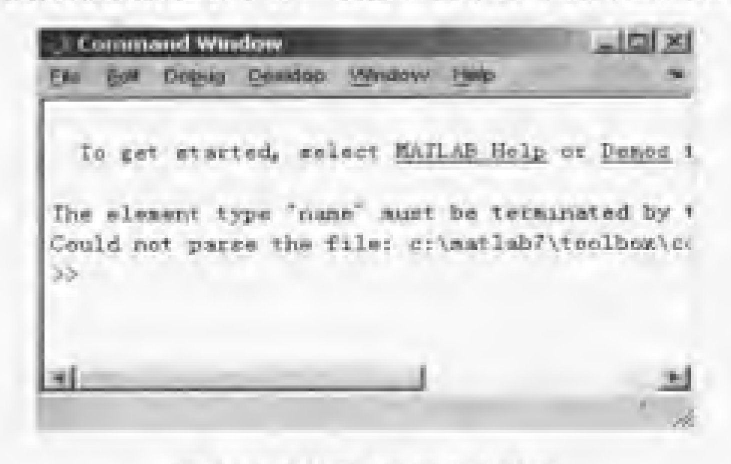


图 2.10　MATLAB 的命令窗口

计算机安装好 MATLAB 之后，双击 MATLAB 图标，就可以进入命令窗口，此时意味着系统处于准备接受命令的状态，可以在命令窗口中直接输入命令语句。

MATLAB 语句形式为：》变量＝表达式。

通过等号将表达式的值赋予变量。当键入回车键时，该语句被执行。语句执行之后，窗口自动显示出语句执行的结果。如果希望结果不被显示，则只要在语句之后加上一个分号即可。此时尽管结果没有显示，但它依然被赋值并在 MATLAB 工作空间中分配了内存。

使用方向键和控制键可以编辑、修改已输入的命令，↑回调上一行命令，↓回调下一行命令。使用“more off”表示不允许分页，“more on”表示允许分页，“more（n）”表示指定每页输出的行数。回车前进一行，空格键显示下一页，“q”结束当前显示。

如果命令语句超过一行或者太长希望分行输入，则可以使用多行命令继续输入。例如，输入下列式子时，可以通过两行输入。

```
S=1-12+13+4+...
9-4-18;
```

2.2.4 MATLAB 工作空间

MATLAB 的工作空间，如图 2.11 所示，它包含了一组可以在命令窗口中调整（调用）的参数，who，whos，clear 是几个常用的工作空间操作的命令，其各自功能描述如下。

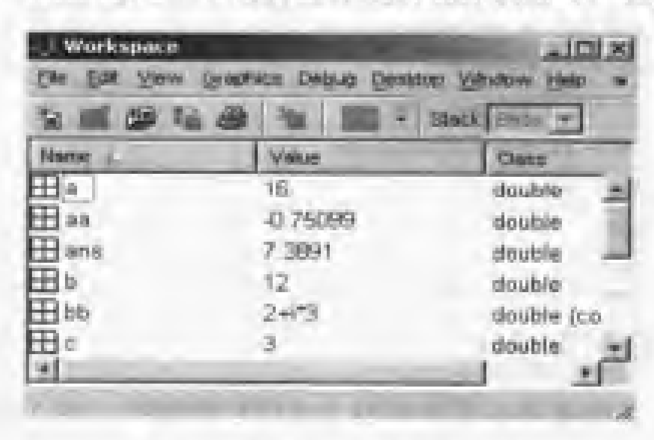


图 2.11　MATLAB 的工作空间

（1）who：显示当前工作空间中所有变量的一个简单列表；

（2）whos：列出变量的大小、数据格式等详细信息；

（3）clear：清除工作空间中所有的变量；

（4）clear 变量名：清除指定的变量。

MATLAB 提供了以下保存和载入工作空间的命令。

1. save filename variables

将变量列表 variables 所列出的变量保存到磁盘文件 filename 中，variables 所表示的变量列表中不能用逗号，各个不同的变量之间只能用空格来分隔。未列出 variables 时，表示将当前工作空间中所有变量都保持到磁盘文件中。默认的磁盘文件扩展名为“.mat”，可以使用“-”定义不同的存储格式（ASCII、V4 等）。

2. load filename variables

将以前用 save 命令保存的变量 variables 从磁盘文件中调入 MATLAB 工作空间。用 load 命令调入的变量，其名称为用 save 命令保存时的名称，取值也一样。variables 所表示的变量列表中，不能用逗号，各个不同的变量之间只能用空格来分隔。未列出 variables 时，表示将磁盘文件中的所有变量都调入工作空间。

3. 退出工作空间

使用 quit 或 exit 命令退出工作空间。

2.2.5　MATLAB 文件管理

MATLAB 提供了一组文件管理命令，包括列文件名、显示或删除文件、显示或改变当前目录等，相关的命令及功能如表 2.1 所示。

表 2.1　MATLAB 常用文件管理命令

命　令	功　能
what	显示当前目录下所有与 MATLAB 相关的文件及它们的路径
dir	显示当前目录下所有的文件
which	显示某个文件的路径
cd path	由当前目录进入 path 目录
type filename	在命令窗口中显示文件 filename
delete filename	删除文件 filename
cd ..	返回上一级目录
cd	显示当前目录

2.2.6　MATLAB 帮助使用

MATLAB 的所有函数都是以逻辑群组方式进行组织的，而 MATLAB 的目录结构就是以这些群组方式来编排的，几个常用的帮助如下。

（1）helpwin：帮助窗口；

（2）helpdesk：帮助桌面，浏览器模式；

（3）lookfor：返回包含指定关键词的项；

（4）demo：打开示例窗口。

MATLAB 还提供了丰富的 help 命令，如表 2.2 所示，在命令窗口中输入相关命令就可以获得相关的帮助。

表 2.2　MATLAB 常用帮助命令

命　令	内　容
help matfun	矩阵函数－数值线性代数
help general	通用命令
help graphics	通用图形函数

续表

命　令	内　容
help elfun	基本的数学函数
help elmat	基本矩阵和矩阵操作
help control	控制系统工具箱函数
help datafun	数据分析和傅里叶变换函数
help ops	操作符和特殊字符
help polyfun	多项式和内插函数
help lang	语言结构和调试
help strfun	字符串函数

2.3　MATLAB 数值计算

MATLAB 是一门计算语言，它的运算指令和语法基于一系列基本的矩阵运算以及它们的扩展运算，它支持的数值元素是复数，这也是 MATLAB 区别于其他高级语言的最大特点之一，它给许多领域的计算带来了极大方便。因此，为了更好地利用 MATLAB 语言的优越性和简捷性，首先需要对 MATLAB 的数值类型、数组矩阵的基本运算、符号运算、关系运算和逻辑运算进行介绍，并给出应用实例，本部分的内容是后面章节的基础。

2.3.1　MATLAB 数值类型

MATLAB 包括 4 种基本数据类型，即双精度数组、字符串数组、元胞数组、构架数组。数值之间可以相互转化，这为其计算功能开拓了广阔的空间。

2.3.1.1　变越与常量

变量是数值计算的基本单元。与 C 语言等其他高级语言不同，MATLAB 语言中的变量无须事先定义，一个变量以其名称在语句命令中第一次合法出现而定义，运算表达式变量中不允许有未定义的变量，也不需要预先定义变量的类型，MATLAB 会自动生成变量，并根据变量的操作确定其类型。

1. MATLAB 变量命名规则

MATLAB 中的变量命名规则如下：

（1）变量名区分大小写，因此 A 与 a 表示的是不同的变量；

（2）变量名以英文字母开始，第一个字母后可以使用字母、数字、下划线，但不能使用空格和标点符号；

（3）变量名长度不得超过 31 位，超过的部分将被忽略；

（4）某些常量也可以作为变量使用，如 i 在 MATLAB 中表示虚数单位，但也可以作为变量使用。

常量是指那些在 MATLAB 中已预先定义其数值的变量，默认的常量如表 2.3 所示。

表 2.3　MATLAB 默认常量

名　　称	说　　明
pi	圆周率
INF（或 inf）	无穷大
NaN（或 nan）	代表不定值（即 0/0）
realman	最大的正实数
realmix	最小的正实数
eps	浮点数的相对误差
i（或 j）	虚数单位，定义为 $\sqrt{-1}$
nargin	函数实际输入参数个数
nargout	函数实际输出参数个数
ANS（或 ans）	默认变量名，以应答最近一次操作运算结果

2. MATLAB 变量的显示

任何 MATLAB 语句的执行结果都可以在屏幕上显示，同时赋值给指定的变量，没有指定变量时，赋值给一个特殊的变量 ans，数据的显示格式由 format 命令控制。format 只影响结果的显示，不影响其计算与存储。MATLAB 总是以双字长浮点数（双精度）来执行所有的运算。如果结果为整数，则显示没有小数；如果结果不是整数，则输出形式有表 2.4 所示的几种形式。

表 2.4　MATLAB 的数据显示格式

格　　式	含　　义
format （short）	短格式（5 位定点数）
format long	长格式（15 位定点数）
format short e	短格式 e 方式
format long e	长格式 e 方式
format bank	2 位十进制格式
format hex	十六进制格式

3. MATLAB 变量的存储

工作空间中的变量可以用 save 命令存储到磁盘文件中。键入命令“save<文件名>”，将工作空间中全部变量存到“<文件名>.mat”文件中去，若省略“<文件名>”则存入文件“matlab.mat”中；命令“save<文件名><变量名集>”将“<变量名集>”指出的变量存入文件“<文件名>.mat”中。

用命令 load 可将变量从磁盘文件读入 MATLAB 的工作空间，其用法为“load<文件名>”，它将“<文件名>”指出的磁盘文件中的数据依次读入名称与“<文件名>”相同的工作空间中的变量中去。若省略“<文件名>”则“matlab.mat”从中读入所有数据。

用 clear 命令可从工作空间中清除现存的变量。

2.3.1.2 字符串

字符是 MATLAB 中符号运算的基本元素，也是文字等表达方式的基本元素，在 MATLAB 中，字符串作为字符数组用单引号（'）引用到程序中，还可以通过字符串运算组成复杂的字符串。字符串数值和数字数值之间可以进行转换，也可以执行字符串的有关操作。

2.3.1.3 元胞数组

元胞是元胞数组（Cell Array）的基本组成部分。元胞数组与数字数组相似，以下标来区分，单元胞数组由元胞和元胞内容两部分组成。用花括号{ }表示元胞数组的内容，用圆括号( )表示元胞元素。与一般的数字数组不同，元胞可以存放任何类型、任何大小的数组，而且同一个元胞数组中各元胞的内容可以不同。

【例 2-1】 元胞数组创建与显示举例。

解：MATLAB 程序代码如下。

```
A(1, 1)={'An example of cell array'};
A(1, 2)={[1 2;3 4]};
A{2, 1}=tf (1, [1, 8]);
A{2, 2}={A(1, 2);'This is an example'};
celldisp(A)
```

元胞数组 A 第 1 行用元胞数组标志法建立一个字符串和一个矩阵；第 2 行用元胞内容编址法，建立一个传递函数和一个由两个元素组成的元胞组，该元胞组分别是矩阵和字符串，最后，用 celldisp 函数显示该元胞数组 A。

2.3.1.4 构架数组

与元胞数组相似，构架数组（Structure Array）也能存放各类数据，使用指针方式传递数值。构架数组由结构变量名和属性名组成，用指针操作符“.”连接结构变量名和属性名。例如，可用 parameter.temperature 表示某一对象的温度参数，用 parameter.humidity 表示某一对象的湿度参数等，因此，该构架数组 parameter 由两个属性组成。

2.3.1.5 对象

面向对象的 MATLAB 语言采用了多种对象，如自动控制中常用的传递函数模型对象（tf object）、状态空间模型对象（ss object）和零极点模型对象（zpk object），一些对象之间可以相互转换，例如可以从传递函数模型对象转化为零极点模型对象，这将在后面具体介绍。

2.3.2 矩阵运算

MATLAB 软件的最大特色是强大的矩阵计算功能，在 MATLAB 软件中，所有的计算都是以矩阵为单元进行的，可见矩阵是 MATLAB 的核心。下面以表格的形式列出

MATLAB 提供的每类矩阵运算的函数，并各举一个实例进行说明，同类函数的用法基本类似，详细的用法及函数内容说明可参考联机帮助。

2.3.2.1　矩阵基本概念

由 m 行 n 列构成的数组 $\boldsymbol{a}$ 称为（$m\times n$）阶矩阵，它总共由（$m\times n$）个元素组成，并按照如下形式排列：

$$\boldsymbol{a}=\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}\rightarrow(m\times n)$$

矩阵元素记为 a_{ij}，其中 i 表示行，j 表示列。当 $m=n$ 时，矩阵 $\boldsymbol{a}$ 称为方阵。当 $i\neq j$ 时，所有的 $a_{ij}=0$，且 $m=n$，得到的矩阵称为对角阵，如下面的矩阵 $\boldsymbol{a}$：

$$\boldsymbol{a}=\begin{pmatrix} a_{11} & \cdots & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{pmatrix}\rightarrow(n\times n)$$

当对角阵的对角线上的元素全为 1 时，称为单位阵，记为 $\boldsymbol{I}$，如下所示：

$$\boldsymbol{I}=\begin{pmatrix} 1 & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & 1 \end{pmatrix}$$

对于 $(m\times n)$ 阶矩阵 $\boldsymbol{w}$，当 $w_{ij}=a_{ji}$ 时，称 $\boldsymbol{w}$ 是 $\boldsymbol{a}$ 的转置矩阵，记为 $\boldsymbol{w}=\boldsymbol{a}'$，如下所示：

$$\boldsymbol{w}=\boldsymbol{a}'=\begin{pmatrix} w_{11}=a_{11} & w_{12}=a_{21} & \cdots & w_{1m}=a_{m1} \\ w_{21}=a_{12} & w_{22}=a_{22} & & \\ \vdots & & \ddots & \\ w_{n1}=a_{1n} & \cdots & & w_{nm}=a_{mn} \end{pmatrix}\rightarrow(n\times m)$$

对于 $\boldsymbol{a}=\begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix}=\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix}\rightarrow(m\times 1)$，称 $\boldsymbol{a}$ 是 m 个元素的列向量，对于 $\boldsymbol{a}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \end{bmatrix}=$ $\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}\rightarrow(1\times n)$ 称 $\boldsymbol{a}$ 是 n 个元素的行向量。

2.3.2.2　矩阵建立与访问

矩阵的表现形式和数组相似，它以左方括号“[”开始，以右方括号“]”结束，每一行元素结束用行结束符号（分号“;”）或回车符分割，每个元素之间用元素分割符号（空格或“,”）分隔。建立矩阵的方法有直接输入矩阵元素、在现有矩阵中添加或删除元素、读取数据文件、采用现有矩阵组合、矩阵转向、矩阵移位及直接建立特殊矩阵等。

【例 2-2】　创建矩阵举例。

解： MATLAB 程序代码如下。

```
>> a=[1 2 3;4 5 6]
```

运行结果是创建了一个 2×3 的矩阵 a，a 的第 1 行由 1、2、3 这 3 个元素组成，第 2 行由 4、5、6 这 3 个元素组成，输出结果如下：

```
a =
        1     2     3
        4     5     6
```

接着输入：

```
>> b=[a; 11, 12, 13]
```

运行结果是创建了一个 3×3 的矩阵 b，b 矩阵是在 a 矩阵的基础上添加一行元素 11、12、13，组成一个 3×3 矩阵，输出结果如下：

```
b =
        1     2     3
        4     5     6
        11    12    13
```

接着输入：

```
>> c=[a', b]
>> b=[a; 11, 12, 13]
```

运行结果是创建了一个 3×5 矩阵 c，c 矩阵是由 a 的转置矩阵和矩阵 b 组合生成的，输出结果如下：

```
c =
        1     4     1     2     3
        2     5     4     5     6
        3     6     11    12    13
```

矩阵元素的访问如下所示。

单个元素的访问：c(3, 5)->13，访问了第 3 行和第 5 列交叉的元素。

整列元素的访问：c(:, 5)->[3, 6, 13]'，访问了第 5 列中的所有元素。

整行元素的访问：c(1, :)->[1, 4, 1, 2, 3]，访问了第 1 行中的所有元素。

整块元素的访问：c(2:3, 3:5)->[4, 5, 6; 11, 12, 13]，访问了一个（2×3）的子块矩阵。

2.3.2.3　特殊矩阵生成

MATLAB 提供了很多个特殊矩阵的生成函数，表 2.5 列出了一些常用的生成函数，关于其他的特殊矩阵生成函数及使用格式，请参见联机帮助。

表 2.5　MATLAB 常用特殊矩阵生成函数

函　数	功能说明
zeros( )	生成元素全为 0 的矩阵
ones( )	全成元素全为 1 的矩阵
rand( )	生成均匀分布随机矩阵

续表

函　数	功 能 说 明
randn( )	生成正态分布随机矩阵
magic( )	生成魔方矩阵
diag( )	生成对角矩阵
triu( )	生成上三角矩阵
tril( )	生成下三角矩阵
eye( )	生成单位矩阵
company( )	生成伴随矩阵
hilb( )	生成 Hilbert 矩阵
vander( )	生成 vander 矩阵
hankel( )	生成 hankel 矩阵
hadamard( )	生成 hadamard 矩阵

【例 2-3】　特殊矩阵生成函数举例。

解：MATLAB 程序代码如下。

```
>> a=[1, 2, 3; 4, 5, 6; 7, 8, 9];
>> b=tril(a)
```

运行结果是生成了 b 矩阵，它是调用下三角矩阵生成函数 tril( )生成的 a 矩阵的下三角矩阵，输出结果如下：

```
b =
     1     0     0
     4     5     0
     7     8     9
```

2.3.2.4　矩阵基本运算

矩阵与矩阵之间可以进行如表 2.6 所示的基本运算。

注意：在进行左除“/”和右除“\”时，两矩阵的维数必须相等。

表 2.6　矩阵基本运算

操 作 符 号	功 能 说 明
+	矩阵加法
−	矩阵减法
*	矩阵乘法
^	矩阵的幂
\	矩阵的右除
/	矩阵的左除
'	矩阵转置
logm( )	矩阵对数运算
expm( )	矩阵指数运算
inv( )	矩阵求逆

【例 2-4】　矩阵基本运算举例。

解：MATLAB 程序代码如下。

```
>> a=[1, 2; 3, 4];
>>b=[3, 5; 2, 9];
>>div1=a/b;
>>div2=b\a
```

两矩阵 a，b 进行了左除和右除运算，输出结果如下：

```
div1 =
    0.2941    0.0588
    1.1176   -0.1765
div2 =
   -0.3529   -0.1176
    0.4118    0.4706
```

2.3.2.5　矩阵函数运算

MATLAB 提供了多种关于矩阵的函数，表 2.7 列出了一些常用的矩阵函数运算。

表 2.7　常用矩阵函数运算

函 数 名	功 能 说 明
rot90( )	矩阵逆时针旋转 90°
flipud( )	矩阵上下翻转
fliplr( )	矩阵左右翻转
flipdim( )	矩阵的某维元素翻转
shiftdim( )	矩阵的元素移位
eig( )	计算矩阵的特征值和特征向量
rank( )	计算矩阵的秩
trace( )	计算矩阵的迹
norm( )	计算矩阵的范数
poly( )	计算矩阵的特征方程的根

【例 2-5】　矩阵函数运算举例。

解：MATLAB 程序代码如下。

```
>> a=[1, 3, 5; 2, 4, 6; 7, 9, 13];
>> [b, c]=eig（a）
```

通过函数 eig( )计算矩阵 a 的特征向量 b 和特征值 c，输出结果如下：

```
b =
   -0.3008   -0.7225    0.2284
   -0.3813   -0.3736   -0.8517
   -0.8742    0.5817    0.4717
```

```
c =
   19.3341         0         0
         0   -1.4744         0
         0         0    0.1403
```

2.3.2.6　矩阵分解运算

矩阵分解常用于方程求根，表 2.8 列出了一些常用的矩阵分解运算。

表 2.8　常用矩阵分解运算函数

函 数 名	功 能 说 明
eig( )	矩阵的特征值分解
qr( )	矩阵的 QR 分解
schur( )	矩阵的 Schur 分解
svd( )	矩阵的奇异值分解
chol( )	矩阵的 Cholesky 分解
lu( )	矩阵的 LU 分解

【例 2-6】　矩阵分解运算函数举例。

解：MATLAB 程序代码如下。

```
>> a=[6, 2, 1; 2, 3, 1; 1, 1, 1];
>> [L, U, P]=lu(a)
```

通过函数 lu( )对矩阵 a 进行 LU 分解，得到上三角阵 U、下三角阵 L、置换矩阵 P，输出结果如下：

```
L =
    1.0000         0         0
    0.3333    1.0000         0
    0.1667    0.2857    1.0000
U =
    6.0000    2.0000    1.0000
         0    2.3333    0.6667
         0         0    0.6429
P =
     1     0     0
     0     1     0
     0     0     1
```

2.4　关系运算和逻辑运算

除了传统的数学运算外，MATLAB 还支持关系运算和逻辑运算。如果你已经有了一

些编程经验，那对这些运算不会陌生。这些操作符和函数的目的是提供求解真/假命题的答案。关系运算和逻辑运算主要用于控制基于真/假命题的各 MATLAB 命令（通常在 M 文件中）的流程或执行次序。

作为所有关系表达式和逻辑表达式的输入，MATLAB 把任何非 0 数值当做真，把 0 当做假。所有关系表达式和逻辑表达式的输出，对于真输出为 1，对于假输出为 0。

MATLAB 为关系运算和逻辑运算提供了关系操作符和逻辑操作符，如表 2.9 和表 2.10 所示。

表 2.9　关系运算符

符　号	功　能
<	小于
<=	小于等于
>	大于
>=	大于等于
==	等于
~=	不等于

表 2.10　逻辑运算符

符　号	功　能
&	逻辑与
|	逻辑或
~	逻辑非

此外，MATLAB 还提供了若干关系运算函数和逻辑运算函数，分别如表 2.11 和表 2.12 所示。

表 2.11　关系运算函数

函 数 名	功　能
all	所有向量为非零元素时为真
any	任一向量为非零元素时为真
xor	逻辑异或运算

表 2.12　逻辑运算函数

函 数 名	功　能
Bitand	位方式的逻辑与运算
Bitor	位方式的逻辑或运算
Bitxor	位方式的逻辑异或运算
Bitcmp	位比较运算
Bitmax	最大无符号浮点整数
Bitshift	将二进制移位运算

位方式的逻辑运算在自动控制系统中应用较少，它在逻辑控制系统中应用较多，故在此不再多做介绍。

2.5 符号运算

MATLAB 提供了符号数学工具箱（Symbolic Math Toolbox），大大增强了 MATLAB 的功能。符号数学工具箱的特点为：

（1）符号数学工具箱适用于广泛的用途，而不是针对一些特殊专业或专业分支。

（2）符号数学工具箱使用字符串来进行符号分析，而不是基于数组的数值分析。

2.5.1 符号运算基础

符号数学工具箱是操作和解决符号表达式的符号数学工具箱（函数）集合，有复合、简化、微分、积分以及求解代数方程和微分方程的工具。另外还有一些用于线性代数的工具，求解逆、行列式、正则形式的精确结果，找出符号矩阵的特征值而没有由数值计算引入的误差。工具箱还支持可变精度运算，即支持符号计算并能以指定的精度返回结果。

1. 符号表达式

符号表达式是代表数字、函数、算子和变量的 MATLAB 字符串，或字符串数组。不要求变量有预先确定的值，符号方程式是含有等号的符号表达式。符号算术是使用已知的规则和给定符号恒等式求解这些符号方程的实践，它与代数和微积分所学到的求解方法完全一样。符号矩阵是数组，其元素是符号表达式。

MATLAB 在内部把符号表达式表示成字符串，与数字变量或运算相区别；否则，这些符号表达式几乎完全像基本的 MATLAB 命令。

2. 符号变量和符号表达式

在 MATLAB 中，用 sym 或 syms 命名符号变量和符号表达式，定义多个符号变量之间用空格分开。例如：

（1）“sym a” 定义了符号变量 a，“syms a b” 定义了符号变量 a 和 b；

（2）“X=sym ('x')” 创建变量 x，“a=sym ('alpha')” 创建变量 alpha；

（3）“syms a b c x ; f=sym (‘a*x^2+b*x+c’)” 创建变量表达式 $f=ax^2+hx+c$ ；

（4）“fcn=sym (‘f(x)’)” 创建函数 f(x)。

2.5.2 常用符号运算

符号变量和数字变量之间可转换，也可以用数字代替符号得到数值。常用的符号运算有代数运算、积分和微分运算、极限运算、级数求和、进行方程求解等。

1. 微分

diff 是求微分最常用的函数，其输入参数既可以是函数表达式，也可以是符号矩阵。常用的格式是：diff（f, x, n），表示 f 关于 x 求 n 阶导数。

【例 2-7】　已知表达式 f=sin（ax），分别对其中的 x 和 a 求导。

解：输入如下 MATLAB 程序代码。

```
>> syms a x
>> f=sin(a*x)
%对 x 求导
>> dfx=diff(f, x)
%对 a 求导
>> dfa=diff(f, a)
```

运行程序，输出结果如下：

```
f =
sin(a*x)
%f 对 x 求导的结果
dfx =
cos(a*x)*a
%f 对 a 求导的结果
dfa =
cos(a*x)*x
```

2. 积分

int 是求积分最常用的函数，其输入参数可以是函数表达式。

常用的格式是：int（f, r, x0, x1）。其中，f 为所要积分的表达式，r 为积分变量，若为定积分，则 x0，x1 为积分上下限。

【例 2-8】　已知表达式 $f = e^{-x^2}$，求对 x 的积分。

解：输入如下 MATLAB 程序代码。

```
>> syms x
>> f=exp(-x^2)
>> int1=int(f, x)
>> int2=int(f, x, -inf, inf)
```

运行程序，输出结果如下：

```
f =
exp(-x^2)
int1 =
1/2*pi^(1/2)*erf(x)
int2 =
pi^(1/2)
```

3. 级数求和

symsum 是用于对符号表达式求和的函数。

常用的格式是：symsum（p, a, b），表示对表达式 p 在[a，b]之间求和。

【例 2-9】 对下列级数求和，$s1=\sum_{k=1}^{\infty}\frac{1}{k^2}$，$s2=\sum_{k=1}^{\infty}\frac{1}{k}$。

解：输入如下 MATLAB 程序代码。

```
>> syms k
>> s1=symsum（1/k^2，1，inf）
>> s2=symsum（1/k，1，inf）
```

运行程序，输出结果如下：

```
s1 =
1/6*pi^2
s2 =
inf
```

2.5.3 控制系统中常用的符号运算

符号数学工具箱为控制理论中常用的积分变换与反变换提供了专用的变换函数与反变换函数，如傅里叶变换 fourier( )、拉普拉斯变换 laplace( )、Z 变换 ztrans( )，以及反变换函数 ifourier( )、ilaplace( )和 iatrans( )。

【例 2-10】 用符号运算计算$G=Ke^{-\frac{t}{T}}$的拉氏变换。

解：MATLAB 程序代码如下。

```
syms K T t
G=K*（exp（-t/T））
% laplace( )：求拉普拉斯变换
Gs=laplace(G)
% simplify( )：对结果进行化简
Gs=simplify（Gs）
```

运行程序，输出结果如下：

```
G =
K*exp(–t/T)
Gs =
K/(s+1/T)
Gs =
K*T/(s*T+1)
```

【例 2-11】 用符号运算计算$\frac{a}{s^2(s+a)}$的脉冲传递函数，采样周期为 T。

解：MATLAB 程序代码如下。

```
syms a T s t k
fs=a/s^2/（s+a）
```

```
% ilaplace( )：求拉普拉斯反变换
ft=ilaplace(fs, t)
% simplify( )：对结果进行化简
ft=simplify(ft)
% subs ( )：进行替换，此处用 k*T 替换 t。
ftt=subs(ft, t, k*T)
% ztrans ( )：求 z 变换
fz=ztrans(ftt)
% simplify( )：对结果进行化简
fz=simplify(fz)
```

运行程序，输出结果如下：

```
fs =
a/s^2/(s+a)
ft =
1/a*(t*a-1+exp(-t*a))
ft =
1/a*(t*a-1+exp(-t*a))
ftt =
1/a*(k*T*a-1+exp(-k*T*a))
fz =
1/a*(T*a*z/(z-1)^2-z/(z-1)+z/exp(-T*a)/(z/exp(-T*a)-1))
fz =
z*(T*a*z*exp(T*a)-T*a+z-z*exp(T*a)-1+exp(T*a))/a/(z-1)^2/(z*exp(T*a)-1)
```

fz 化简后为：$\dfrac{Tz}{(z-1)^2}-\dfrac{z(1-\mathrm{e}^{-aT})}{a(z-1)(z-\mathrm{e}^{-aT})}$。

2.6　MATLAB 常用绘图命令

MATLAB 提供了强大的图形用户界面，在许多应用中，常常要用绘图功能来实现数据的显示和分析，包括二维图形和三维图形。在控制系统仿真中，也常常用到绘图，如绘制系统的响应曲线、根轨迹或频率响应曲线等。

MATLAB 提供了丰富的绘图功能，在命令窗口中输入“help graph2d”可得到所有画二维图形的命令；输入“help graph3d”可得到所有画三维图形的命令。

下面主要介绍常用的二维图形命令的使用方法，三维图形命令的使用方法与此类似。

1. 基本的绘图命令

plot(x1, y1, option1, x2, y2, option2, …)：x1, y1 给出的数据分别为 x, y 轴坐标值，

option1 为选项参数，以逐点连折线的方式绘制 1 个二维图形；同时类似地绘制第二个二维图形。这是 plot 命令的完全格式，在实际应用中可以根据需要进行简化。比如 plot(x, y)；plot(x, y, option)，选项参数 option 定义了图形曲线的颜色、线型及标示符号，它由一对单引号括起来。

2. 选择图像命令

figure(1)；figure(2)；…；figure(n)，它用来打开不同的图形窗口，以便绘制不同的图形。

3. 在图形上添加或删除栅格命令

grid on：在所画出的图形坐标中加入栅格；grid off：除去图形坐标中的栅格

4. 图形保持或覆盖命令

hold on：把当前图形保持在屏幕上不变，同时允许在这个坐标内绘制另外一个图形。hold off：使新图覆盖旧图。

【例 2-12】 plot 绘图命令举例。

解：MATLAB 程序代码如下。

```
%关闭打开了的所有图形窗口
close all
%清屏命令
clc
%清除工作空间中所有变量
clear
%定义时间范围
t=[0:pi/20:8*pi];
y=cos (t);
plot(t, y, 'b:+')
% r 表示统的颜色为红色,
%此外 y(黄色) g (绿色) b (蓝色)
% w(白色) k (黑色)m (紫色) c (青色)
% -.表示线型为点划线,
%此外 –(实线) : (虚线)——(破折线)
% *表示标示符号为星号,
%此外 +(正号) o (圆形字母)
% x (交叉字母) square (方形) . (点)
```

运行后，输出结果如图 2.2 所示，当需要在同一图上画多条曲线时，可以通过选择不同的 option 来加以区别。

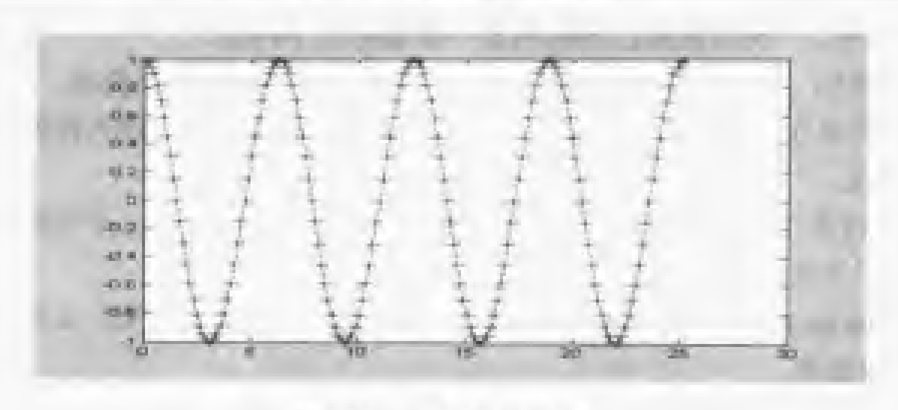


图 2.12　例 2-12 的输出图

【例 2-13】　绘图命令使用举例。

在程序头部设定断点，然后单步执行，可以看出所画曲线及栅格的变化。

解：MATLAB 程序代码如下。

```
%关闭打开了的所有图形窗口
close all
%清屏命令
clc
%清除工作空间中所有变量
clear
%定义时间范围
t=[0:pi/20:9*pi];
%允许在同一坐标系下绘制不同的图形
hold on
plot(t, sin(t), 'r:*')
plot(t, cos(t))
plot(t, -cos(t), 'k')
%在所画出的图形坐标中添加栅格，注意用在 plot 之后
grid on
%剔除栅格
grid off
%覆盖旧图
hold off
plot(t, -sin(t))
```

（1）设定轴范围的命令。

axis([xmin xmax ymin ymax]), axis('equal')：将 x 坐标轴和 y 坐标轴的单位刻度大小调整为一样。

（2）文字标示命令。

text(x, y, '字符串')：在图形的指定坐标位置（x, y）处标示单引号括起来的字符串。

gtext('字符串')：利用鼠标在图形的某一位置标示字符串。

title('字符串')：在所画图形的最上端显示说明该图形标题的字符串。

xlabel('字符串')，ylabel('字符串')：设置 x, y 坐标轴的名称。输入特殊的文字需要用反斜杠（\）开头。

legend('字符串 1', '字符串 2', …, '字符串 n')：在屏幕上开启一个小视窗，然后依据绘图命令的先后次序，用对应的字符串区分图形上的线。

（3）subplot(m, n, k)：分割图形显示窗口，m 表示上下分割个数，n 表示左右分割个数，k 表示子图编号。

（4）半对数坐标绘制命令。

semilogx：绘制以 x 轴为对数坐标（以 10 为底）、y 轴为线性坐标的半对数坐标图形。

semilogy：绘制以 y 轴为对数坐标（以 10 为底）、x 轴为线性坐标的半对数坐标图形。

（5）常用的应用型绘图指令，可用于数值统计分析或离散数据处理。

bax(x, y)：绘制对应于输入 x 和输出 y 的高度条形图。

hist(y, x)：绘制 x 在以 y 为中心的区间中分布的个数条形图。

stairs(x, y)：绘制 y 对应于 x 的梯形图。

stem(x, y)：绘制 y 对应于 x 的散点图。

需要注意的是，对于图形的属性编辑同样可以在图形窗口上直接进行，但图形窗口关闭之后编辑结果不会保存。

2.7 MATLAB 程序设计

2.7.1 MATLAB 程序类型

MATLAB 程序类型包括三种：一种是在命令窗口下执行的脚本 M 文件；另外一种是可以存取的 M 文件，即程序文件；最后一种是函数（function）文件。脚本 M 文件和程序文件中的变量都将保存在工作区中，这一点与函数文件是截然不同的。

1. 脚本 M 文件

脚本 M 文件也称命令文件，它在命令窗口中输入并执行；没有输入参数，也不返回输出参数，只是一些命令行的组合；脚本 M 文件可对工作空间中的变量进行操作，也可生成新的变量；脚本 M 文件运行结束后，脚本 M 文件产生的变量仍将保留在工作空间中，直到关闭 MATLAB 或用相关命令删除。

2. 程序文件

以.m 格式进行存取，包含一连串的 MATLAB 指令和必要的注解。需要在工作空间中创建并获取变量，也就是说处理的数据为命令窗口中的数据，没有输入参数，也不会返回参数。程序运行时只需在工作空间中键入其名称即可。

在 MATLAB 命令窗口中选定“File”菜单“New”” 选项“M-file”即可建立 M 文件。

也可选定“Edit”菜单建立 M 文件，选定“Save”选项即可保存文件。

选定 MATLAB 命令窗口中的“Edit”菜单可利用键盘编辑键对 M 文件进行全屏幕编辑。M 文件以 ASCII 编码形式存储，在命令窗口中直接键入文件名就可执行 M 文件。

3. 函数文件

与在命令窗口中输入命令一样，函数接受输入参数后执行并输出结果。用 help 命令可以显示它的注释说明。函数文件具有标准的基本结构。

（1）函数定义行（关键字 function）。

```
function[out1, out2, ..]=filename(in1, in2, ..)
```

输入和输出（返回）的参数个数分别由 nargin 和 nargout 两个 MATLAB 保留的变量给出。

（2）第一行帮助行，即 H1 行，以%开头，作为 lookfor 指令搜索的行。

（3）函数体说明及有关注解以%开头，用以说明函数的作用及有关内容，作为 M 文件的帮助信息。如果不希望显示某段信息，可在它的前面加空行。

（4）函数体语句：除返回和输入变量这些在 function 语句中直接引用的变量以外，函数体内使用的所有变量都是局部变量，即在该函数返回之后，这些变量会自动在 MATLAB 的工作空间中清除。如果希望这些中间变量成为在整个程序中都起作用的变量，则可以将它们设置为全局变量。

2.7.2 MATLAB 程序流程控制

MATLAB 程序有顺序、分支、循环等程序结构以及子程序结构。

1. 顺序程序结构

顺序程序结构的程序从程序的首行开始，逐行顺序往下执行，直到程序最后一行，大多数简单的 MATLAB 程序采用这种程序结构。

2. 分支程序结构

分支程序结构的程序根据执行条件满足与否，确定执行方向。在 MATLAB 中，通过 if-else-end 结构、while 结构、switch-case-otherwise 结构来实现。

（1）if，else，elseif 语句

if 条件语句用于选择结构。其格式有以下情况：

```
① if    逻辑表达式
        执行语句
   end
② if    逻辑表达式
        执行语句 1
  else
```

```
        执行语句 2
    end
```

如果逻辑表达式的值为真，则执行语句 1，然后跳过语句 2，向下执行；如果为假，则执行语句 2，然后向下执行。

```
③ if      逻辑表达式 1
            执行语句 1
  elseif   逻辑表达式 2
            执行语句 2
  end      …
```

如果逻辑表达式 1 的值为真，则执行语句 1；如果为假，则判断逻辑表达式 2，如果为真，则执行语句 2，否则向下执行。

if 条件语句可以嵌套使用，但是，必须注意 if 语句和 end 语句成对出现。

【例 2-14】 if 条件语句使用举例。

解：MATLAB 程序代码如下。

```
n=input('n=')
%判断输入数的正负性
if n<=0    A= 'negative'
%判断输入是否为空
elseif   isempty(n)==1
A= 'empty'
%除 2 取余数，判断奇偶性
elseif   rem(n, 2)==0
A= 'even'
else
A= 'odd'
end
```

输出结果如下：

```
n =
 []
A =
empty
n =
4
A =
even
n =
-4
A =
negative
```

（2）switch 语句

基本格式：

```
switch    表达式（%可以是标量或字符串）
          case      值 1
                     语句 1
          case      值 2
                     语句 2
                ….
           otherwise
           语句 3
           end
```

表达式的值和哪种情况（case）的值相同，就执行哪种情况中的语句，如果不同，则执行 otherwise 中的语句。格式中也可以不包括 otherwise，这时如果表达式的值与列出的各种情况都不相同，则继续向下执行。

3. 循环程序结构

循环程序结构包括一个循环变量，循环变量从初始值开始计数，每循环一次就执行一次循环体内的语句，执行后，循环变量以一定的规律变化，然后再执行循环体内语句，直到循环变量大于循环变量的终止值为止。

常用的循环有 while 和 for 循环。while 循环和 for 循环的区别在于：while 循环结构的循环体被执行的次数不是确定的，而 for 结构中循环体的执行次数是确定的。

（1）for 循环语句

for 语句使用较为灵活，一般用于循环次数已经确定的情况，其格式为：

```
for   循环变量=起始值：步长：终止值
      循环体
end
```

步长默认值为 1，可以在正实数或负实数范围内任意指定。对于正数，循环变量的值大于终止值时，循环结束；对于负数，循环变量的值小于终止值时，循环结束。循环结构可以嵌套使用。书写格式不必太过于拘泥，在 Editor 编辑器中会自动进行处理。

for 语句允许嵌套。在程序里，每一个“for”关键字必须和一个“end”关键字配对，否则程序执行时出错。

【例 2-15】　for 循环语句使用举例。

解：MATLAB 程序代码如下。

```
%计算出 1~4 的乘法表
for n = 1:4
for  m=1:n
r（n，m）=m*n
end
end
```

输出结果如下：

```
r =
      1     0     0     0
      2     4     0     0
      3     6     9     0
      4     8    12    16
```

（2）while 循环语句

while 语句一般用于事先不能确定循环次数的情况，其基本格式为。

```
while   表达式
        循环体
end
```

若表达式为真，则执行循环体的内容，执行后再判断表达式是否为真；若不为真，则跳出循环体，向下继续执行。在 while 语句的循环中，可用 break 语句退出循环。

【例 2-16】　while 循环语句使用举例。

解：MATLAB 程序代码如下。

```
%计算 2000 以内的 fibnacci 数
f(1)=1
f(2)=1
i=1
while f(i)+f(i+1)<2000
f(i+2)=f(i)+f(i+1)
i=i+1
end
f
```

程序的输出结果如下：

```
i =
    16
f =
Columns 1 through 8
1  1  2  3  5  8  13  21
Columns 9 through 16
34  55  89  144  233  377  610  987
Column 17
1597
```

2.7.3　MATLAB 程序基本设计原则

MATLBA 程序的基本设计原则如下所述：

（1）%后面的内容是程序的注解，要善于运用注解使程序更具可读性。

（2）养成在主程序开头用 clear 指令清除变量的习惯，以消除工作空间中其他变量对程序运行的影响，但注意在子程序中不要用 clear。

（3）参数值要集中放在程序的开始部分，以便维护。要充分利用 MATLAB 工具箱提供的指令来执行所要进行的运算，在语句行之后输入分号使其及中间结果不在屏幕上显示，以提高执行速度。

（4）input 指令可以用来输入一些临时的数据；对于大量参数，则通过建立一个存储参数的子程序，在主程序中通过子程序的名称来调用。

（5）程序尽量模块化，即采用主程序调用子程序的方法，将所有子程序合并在一起来执行全部的操作。

（6）充分利用 Debugger 来进行程序的调试（设置断点、单步执行、连续执行），并利用其他工具箱或图形用户界面（GUI）的设计技巧，将设计结果集成到一起。

（7）设置好 MATLAB 的工作路径，以便程序运行。

（8）MATLAB 程序的基本组成结构如下所示：

```
%说明
清除命令：清除 workspace 中的变量和图形（clear，close）
定义变量：包括全局变量的声明及参数值的设定
逐行执行命令：指 MATLAB 提供的运算指令或工具箱提供的专用命令
…
…
…
```

控制循环：包含 for，if then，switch，while 等语句

逐行执行命令

...

...

end

绘图命令：将运算结果绘制出来

当然，更复杂的程序还需要调用子程序，或者与 Simulink 及其他应用程序相结合。

第 3 章 仿真集成环境 Simulink

3.1 引 言

1990 年，MathWorks 软件公司为 MATLAB 提供了新的控制系统模型化图形输入与仿真工具，并命名为 SIMULAB，该工具很快就在控制工程界获得了广泛的认可，使得仿真软件进入了模型化图形组态阶段。1992 年正式将该软件更名为 Simulink。

Simulink 的出现给控制系统分析与设计带来了福音。它有两个主要功能：Simu（仿真）和 Link（连接），即该软件可以利用鼠标在模型窗口上绘制出所需要的控制系统模型，然后利用 Simulink 提供的功能来对系统进行仿真和分析。

在实际工程中，控制系统的结构往往很复杂，如果不借助专用的系统建模软件，则很难准确地把一个控制系统的复杂模型输入计算机，对其进行进一步的分析与仿真。因此，熟悉掌握 Simulink 对于一个当代从事自动控制方面工作的人来说是非常必要的。

通过本章使读者对 Simulink 的基本模块和功能有一个全面了解，并能熟练 Simulink 的基本操作，为使用 simulink 进行控制系统仿真奠定基础。

3.2 Simulink 的使用

Simulink 是 MATLAB 软件的扩展，它是实现动态系统建模和仿真的一个软件包，它与 MATLAB 语言的主要区别在于、它与用户交互模口是基于 Windows 的模型化图形输入的，从而使得用户可以把更多的精力投入到系统模型的构建而非语言的模程上。

所谓模型化图形输入是指 Simulink 提供了一些按功能分类的基本系统模块，用户只需要知道这些模块的输入、输出及模块的功能，而不必考察模块内部是如何实现的，通过对这些基本模块的调用，再将它们连模起来就可以构成所需要的系统模型（以.mdl 文件进行存取），进而进行仿真与分析。

Simulink 的最新版本是 Simulink 6.0（包含在 MATLAB 7.0 里），MATLAB 6.5 里的版本为 5.0 版，它们的基本功能相差不大，一些主要的变化在第 2 章已有介绍。

3.2.1 Simulink 启动

Simulink 的启动有两种方式，一种是启动 MATLAB 后，单击 MATLAB 主窗口的快捷按钮来打开 Simulink Library Browser 窗口，如图 3.1 所示。

另一种是在 MATLAB 命令窗口中输入“Simulink”，结果是在桌面上出现一个称为 Simulink Library Browser 的窗口，在这个窗口中列出了模功能分类的各种模块的名称。



图 3.1　Simulink 模块库浏览界面

在 MATLAB 命令窗口中输入"simulink3"，结果是在桌面上出现一个用图标形式显示的 Library :simulink3 的 Simulink 模块库窗口，如图 3.2 所示。这两种模块库窗口界面只是不同的显示形式，用户可以根据个人喜好进行选用，一般第 2 种窗口更直观、形象，适合初学者，但使用时会打开太多的子窗口。

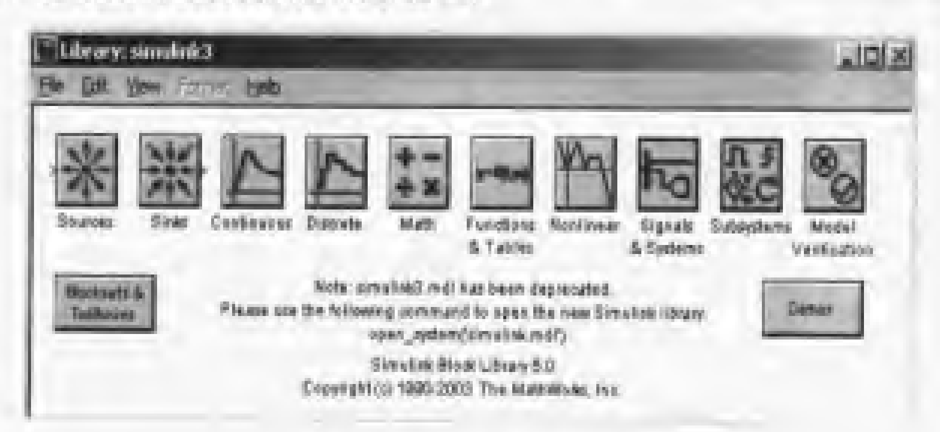


图 3.2　Simulink 模块库窗口

Simulink 启动后，便可打开如图 3.3 所示的 Simulink 的仿真编辑窗口，用户此时就可以开始编辑自己的仿真程序了。



图 3.3　Simulink 仿真编辑窗口

3.2.2　Simulink 仿真设置

在编辑好仿真程序后，应设置仿真操作参数，以便进行仿真。单击 Simulation 菜单下面的 Configuration Parameters 项或者直接按快捷键“Ctrl+E”，便弹出如图 3.4 所示的设置界面，它包括仿真器参数（Solver）设置、工作空间数据导入/导出（Data Import/Export）设置、优化（Optimization）设置、诊断参数（Diagnostics）设置、硬件实现（Hardware Implementation）设置、模型引用（Model Referencing）设置和实时代码生成工具（Real-Time Workshop）设置。



图 3.4　Simulink 设置窗口

3.2.2.1　仿真器参数设置

仿真器参数设置的界面如图 3.5 所示，它可用于仿真开始时间、仿真结束时间、解法器及输出项等的选择。

1. 仿真时间

这里所指的时间概念与真实的时间并不一样，只是计算机仿真中对时间的一种表示，比如 10 秒的仿真时间，如果采样步长定为 0.1，则需要执行 100 步，若把步长减小，则采样点数增加，那么实际的执行时间就会增加。一般仿真开始时间设为 0，而结束时间则视不同的情况进行选择。总的说来，执行一次仿真要耗费的时间依赖很多因素，包括模型的复杂程度、解法器及其步长的选择、计算机时钟的速度等。

图 3.5　仿真器参数设置窗口

2. 仿真步长模式

用户在 Type 后面的第一个下拉选项框中指定仿真的步长选取方式，如图 3.6 所示，可供选择的有 Variable-step（变步长）和 Fixed-step（固定步长）方式。选择变步长模式则可以在仿真过程中改变步长，提供误差控制和过零检测选择。固定步长模式则可以在仿真过程中提供固定的步长，不提供误差控制和过零检测。用户还可以在第二个下拉选项框中选择对应模式下仿真所采用的算法。

图 3.6　仿真类型设置窗口

用户在 Solver 后面的下拉选项中选择变步长模式解法器，如图 3.7 所示，变步长模式解法器有：ode45，ode23，ode113，ode15s，ode23s，ode23t，ode23tb 和 discrete。下面简要概述一下这些解法器的含义。

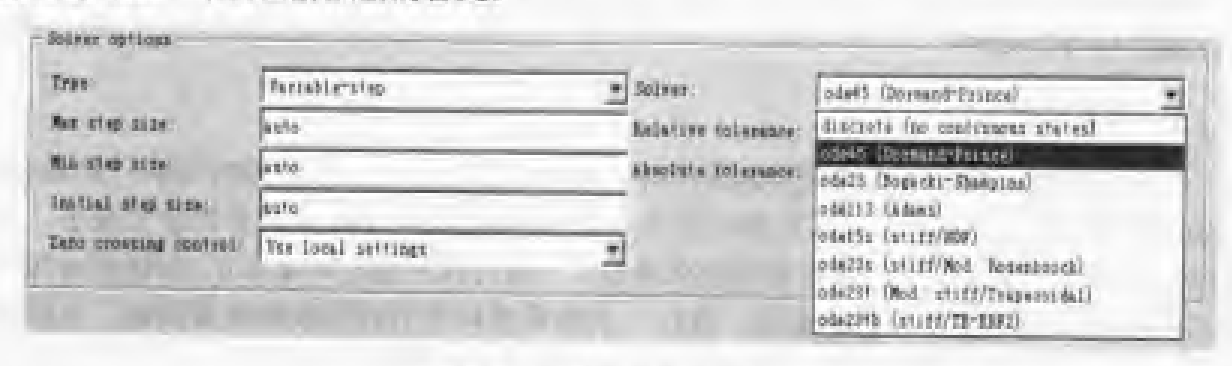


图 3.7　解法器参数设置窗口

（1）ode45：默认值，表示四/五阶龙格－库塔法，适用于大多数连续或离散系统，但不适用于刚性（stiff）系统。它是单步解法器，即在计算 $y(t_n)$ 时，它仅需要最近处理时刻的结果 $y(t_{n-1})$。一般来说，面对一个仿真问题最好首先试试 ode45。

（2）ode23：表示二/三阶龙格－库塔法，它在误差限要求不高和求解的问题不太难的情况下，可能会比 ode45 更有效。它也是一个单步解法器。

（3）ode113：表示一种阶数可变的解法器，它在误差容许要求严格的情况下通常比 ode45 有效。ode113 是一种多步解法器，即在计算当前时刻输出时，它需要以前多个时刻的解。

（4）ode15s：表示一种基于数字微分公式的解法器（NDFs），它也是一种多步解法器。适用于刚性系统，当用户估计要解决的问题是比较困难的、不能使用 ode45 或者即使使用效果也不好时，就可以用 ode15s。

（5）ode23s：表示一种单步解法器，专门应用于刚性系统，在弱误差允许下的效果优于 ode15s。它能解决某些 ode15s 所不能有效解决的 stiff 问题。

（6）ode23t：表示梯形规则的一种自由插值实现。这种解法器适用于求解适度 stiff 而用户又需要一个无数字振荡的解法器的情况。

（7）ode23tb：表示 TR-BDF2 的一种实现，TR-BDF2 是具有两个阶段的隐式龙格－库塔公式。

（8）discrete：当 Simulink 检查到模型没有连续状态时使用它。

固定步长模式解法器有：ode5，ode4，ode3，ode2，ode1 和 discrete。

（1）ode5：默认值，是 ode45 的固定步长版本，适用于大多数连续或离散系统，不适用于刚性系统。

（2）ode4：表示四阶龙格－库塔法，具有一定的计算精度。

（3）ode3：表示固定步长的二/三阶龙格－库塔法。

（4）ode2：表示改进的欧拉法。

（5）ode1：表示欧拉法。

（6）discrete：表示一种实现积分的固定步长解法器，它适合于离散无连续状态的系统。

3. 步长参数

对于变步长模式，用户可以设置最大的和推荐的初始步长参数，默认情况下，步长自动确定，用 auto 值表示。

（1）Maximum step size（最大步长参数）：决定解法器能够使用的最大时间步长，它的默认值为“仿真时间/50”，即整个仿真过程中至少取 50 个取样点，但这样的取法对于仿真时间较长的系统则可能带来取样点过于稀疏的问题，继而使仿真结果失真。一般建议对于仿真时间不超过 15s 的采用默认值即可，对于超过 15s 的每秒至少保证 5 个采样点，对于超过 100s 的，每秒至少保证 3 个采样点。

（2）Initial step size（初始步长参数）：一般建议使用 auto 默认值。

4. 仿真精度定义（对于变步长模式）

（1）Relative tolerance（相对误差）：指误差相对于状态的值，是一个百分比，默认值为 1e–3，表示状态的计算值要精确到 0.1%。

（2）Absolute tolerance（绝对误差）：表示误差值的门限，或者是在状态值为零的情况下可以接受的误差。如果它被设成了 auto，那么 Simulink 为每一个状态设置初始绝对误差为 1e–6。

5. Mode（固定步长模式选择）

（1）Multitasking：选择这种模式时，当 Simulink 检测到模块间非法的采样速率转换时系统会给出错误提示。所谓的非法采样速率转换指两个工作在不同采样速率的模块之间的直接连接。在实时多任务系统中，如果任务之间存在非法采样速率转换，那么就有可能出现一个模块的输出在另一个模块需要时却无法利用的情况。通过检查这种转换，Multitasking 将有助于用户建立一个符合现实的多任务系统的有效模型。使用速率转换模块可以减少模型中的非法速率转换。Simulink 提供了两个这样的模块：unit delay 模块和 zero-order hold 模块。对于从慢速率到快速率的非法转换，可以在慢输出端口和快输入端口插入一个单位延时（unit delay）模块。对于快速率到慢速率的转换，则可以插入一个零阶采样保持器（zero-order hold）。

（2）Singletasking：这种模式不检查模块间的速率转换，它在建立单任务系统模型时非常有用，在这种系统中不存在任务同步问题。

（3）Auto：选择这种模式时，Simulink 会根据模型中模块的采样速率是否一致，自动决定切换到 Multitasking 模式或 Singletasking 模式。

6. 输出选项

（1）Refine output：这个选项可以理解成精细输出，其意义是在仿真输出太稀松时，Simulink 会产生额外的精细输出，这一点就像插值处理一样。用户可以在 refine factor 设置仿真时间步间插入的输出点数。产生更光滑的输出曲线，改变精细因子比减小仿真步长更有效。精细输出只能在变步长模式中才能使用，并且在 ode45 效果最好。

（2）Produce additional output：它允许用户直接指定产生输出的时间点。一旦选择了该项，则在它的右边出现一个 output times 编辑框，在这里用户指定额外的仿真输出点，它既可以是一个时间向量，也可以是表达式。与精细因子相比，这个选项会改变仿真的步长。

（3）Produce specified output only：它的意思是让 Simulink 只在指定的时间点上产生输出。为此解法器要调整仿真步长以使之和指定的时间点重合。这个选项在比较不同的仿真时可以确保它们在相同的时间输出。

3.2.2.2　工作空间数据导入/导出设置

工作空间数据导入/导出（Data Import/Export）设置的界面如图 3.8 所示，它主要在 Simulink 与 MATLAB 工作空间交换数值时进行有关选项设置，包括 Load from workspace、Save to workspace 和 Save option 三个选择项。

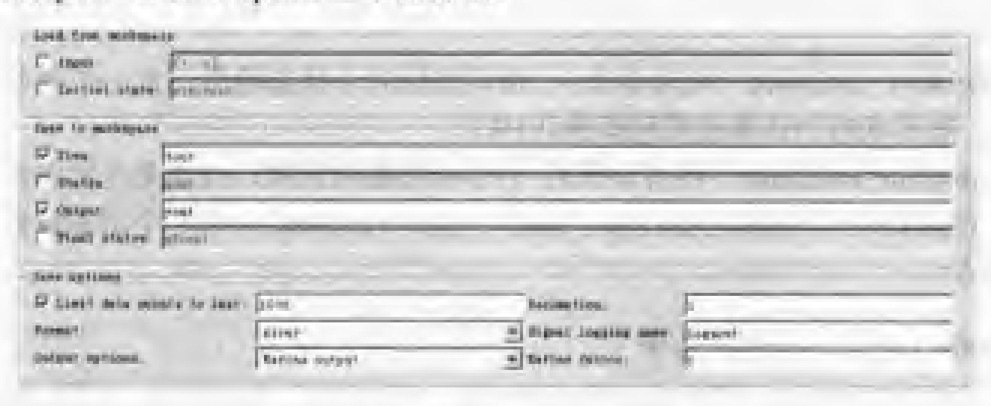


图 3.8　工作空间数据导入/导出设置窗口

（1）Load from workspace：选中前面的复选框即可从 MATLAB 工作空间获取时间和输入变量，一般时间变量定义为 t，输入变量定义为 u。Initial state 用来定义从 MATLAB 工作空间获得的状态初始值的变量名。

（2）Save to workspace：用来设置存在 MATLAB 工作空间的变量类型和变量名，选中变量类型前的复选框使相应的变量有效。一般存往工作空间的变量包括输出时间向量（Time）、状态向量（States）和输出变量（Output）。Final states 用来定义将系统稳态值存往工作空间时所用的变量名。

（3）Save options：用来设置存往工作空间的有关选项。Limit rows to last 用来设定 Simulink 仿真结果最终可存往 MATLAB 工作空间的变量的规模，对于向量而言即其维数，对于矩阵而言即其秩；Decimation 设定了一个亚采样因子，它的默认值为 1，也就是对每一个仿真时间点产生值都保存，若为 2 则是每隔一个仿真时刻才保存一个值。Format 用来说明返回数据的格式，包括矩阵（matrix）、结构体（struct）及带时间的结构体（struct with time）。

3.2.2.3　诊断参数设置

诊断参数（Diagnostics）设置的窗口如图 3.9 所示，它包括采样时间（Sample Time）、数据完整性（Data Integrity）、转换（Conversion）、连接（Connectivity）、兼容性

（Compatibility）和模型引用（Model Referencing）这几个子项的诊断。用户可以设置为当 Simulink 检查到这些子项事件时应做处理，主要包括是否进行一致性检验、是否禁用过零检测、是否禁止复用缓存、是否进行不同版本的 Simulink 的检验等几项。

3.2.2.4　实时代码生成工具设置

仿真参数设置窗口还包括 Real-Time Workshop 设置，如图 3.10 所示，主要用于与 C 语言编辑器的交换，通过它可以直接从 Simulink 模型生成代码并且自动建立可以在不同环境下运行的程序，这些环境包括实时系统和单机仿真。

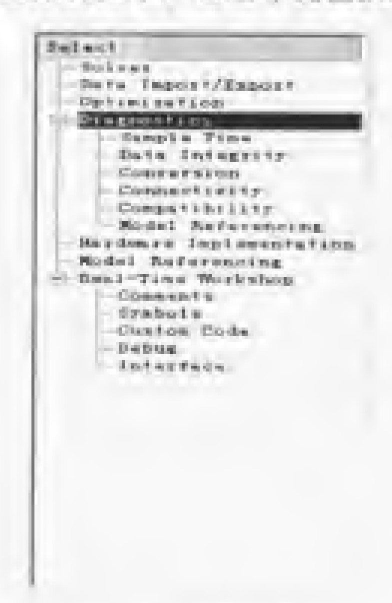


图 3.9　诊断参数设置窗口

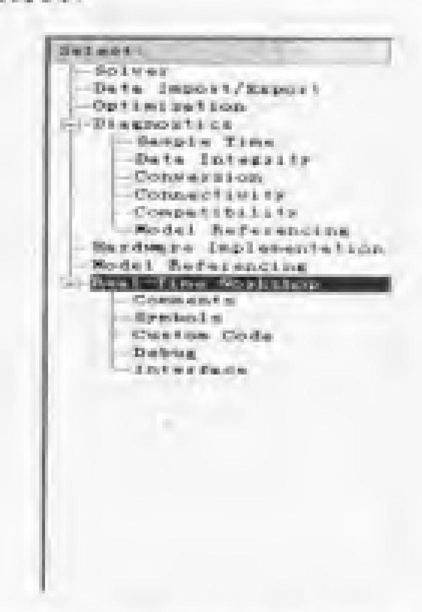


图 3.10　实时代码生成工具设置窗口

3.2.2.5　其他设置

仿真参数设置窗口还包括优化（Optimization）、硬件实现（Hardware Implementation）和模型引用（Model Referencing）设置。

Optimization 设置窗口如图 3.11 所示，用于设置模拟仿真的优化参数。

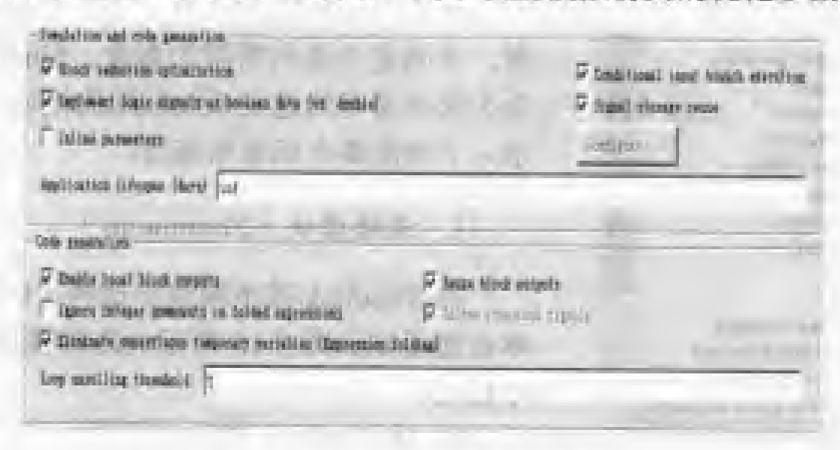


图 3.11　Optimization 设置窗口

Hardware Implementation 设置窗口如图 3.12 所示，包括 Microprocessor 和 Unconstrained Integer size 两个选项，用于设置仿真硬件特性。

Model Referencing 设置窗口如图 3.13 所示，用于设置模型引用的有关参数。

图 3.12　Hardware Implementation 设置窗口



图 3.13　Model Referencing 设置窗口

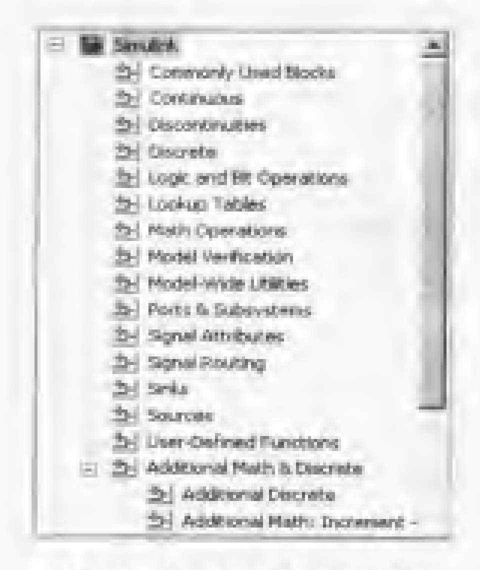


图 3.14　Simulink 模块浏览窗口

3.2.3　Simulink 模块库简介

在进行系统动态仿真之前，应绘制仿真系统框图，并确定仿真所需用的参数。Simulink 模块库包含大部分常用的建立系统框图的模块，如图 3.14 所示。下面简要介绍常用模块。

1．连续模块（Continuous）

在 Simulink 基本模块中选择“Continuous”后，单击便看到如图 3.15 所示的连续模块，它包括以下子模块。

- Derivative：输入信号微分；
- Integrator：输入信号积分；

- State-Space：状态空间系统模型；
- Transfer-Fcn：传递函数模型；
- Transport Delay：输入信号延迟一个固定时间再输出；
- Variable Transport Delay：输入信号延迟一个可变时间再输出；
- Zero-Pole：零极点模型。

2. 非连续模块（Discontinuous）

在 Simulink 基本模块中选择 Discontinuous 后，单击便看到如图 3.16 所示的非连续模块，它包括以下子模块。



图 3.15　连续模块



图 3.16　非连续模块

- Backlash：间隙非线性；
- Coulomb&Viscous Friction：库仑和黏度摩擦非线性；
- Dead Zone：死区非线性；
- Dead Zone Dynamic：动态死区非线性；
- Hit Crossing：冲击非线性；
- Quantizer：量化非线性；
- Rate Limiter：静态限制信号的变化速率；
- Rate Limiter Dynamic：动态限制信号的变化速率；
- Relay：滞环比较器，限制输出值在某一范围内变化；
- Saturation：饱和输出，让输出超过某一值时能够饱和；
- Saturation Dynamic：动态饱和输出；
- Wrap To Zero：环零非线性。

3. 离散模块（Discrete）

在 Simulink 基本模块中选择“Discrete”后，单击便看到如图 3.17 所示的离散模块。

它包括以下子模块。





图 3.17　离散模块

- Difference：差分环节；
- Discrete Derivative：离散微分环节；
- Discrete Filter：离散滤波器；
- Discrete State-Space：离散状态空间系统模型；
- Discrete Transfer-Fcn：离散传递函数模型；
- Discrete Zero-Pole：以零极点表示的离散传递函数模型；
- Discrete-time Integrator：离散时间积分器；
- First-Order Hold：一阶保持器；
- Integer Delay：整数被延迟；
- Memory：输出本模块上一步的输入值；
- Tapped Delay：延迟；
- Transfer Fcn First Order：离散一阶传递函数；
- Transfer Fcn Lead or Lag：传递函数；
- Transfer Fcn Real Zero：离散零点传递函数；
- Unit Delay：一个采样周期的延迟
- Weighted Moving Average：权值移动平均模型；
- Zero-Order Hold：零阶保持器。

4. 逻辑和位操作模块（Logic and Bit Operations）

在 Simulink 基本模块中选择 Logic and Bit Operations 后，单击便看到如图 3.18 所示的逻辑和位操作模块，它包括以下子模块：



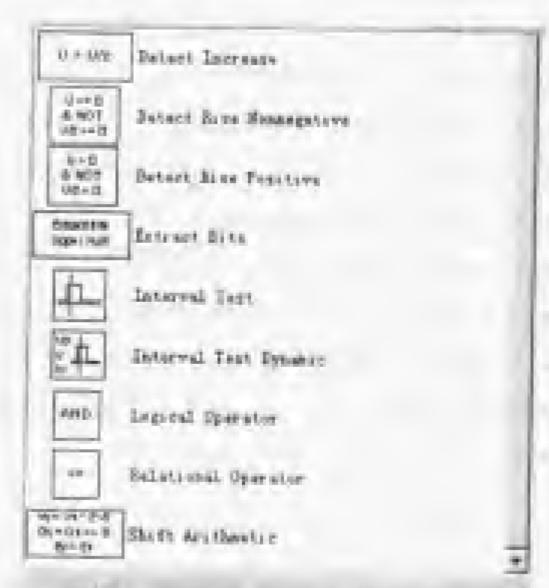


图 3.18　逻辑和位操作模块

- Bit Clear：位清零；
- Bit Set：位置位；
- Bitwise Operator：逐位操作；
- Combinatorial Logic：组合逻辑；
- Compare To Constant：和常量比较；
- Compare To Zero：和零比较；
- Detect Change：检测跳变；
- Detect Decrease：检测递减；
- Detect Fall Negative：检测负下降沿；
- Detect Fall Nonpositive：检测非负下降沿；
- Detect Increase：检测递增；
- Detect Rise Nonnegative：检测非负上升沿；
- Detect Rise Positive：检测正上升沿；
- Extract Bits：提取位；
- Interval Test：检测开区间；
- Interval Test Dynamic：动态检测开区间；
- Logical Operatior：逻辑操作符；
- Relational Operator：关系操作符；
- Shift Arithmetic：移位运算。

5．查找表模块（Lookup Table）



图 3.19　查找表模块

在 Simulink 基本模块中选择 “Lookup Table” 后，单击便看到如图 3.19 所示的查找表模块，它包

括以下子模块。

- Cosine：余弦函数查询表；
- Direct Lookup Table (n-D) ：n 个输入信号的查询表（直接匹配）；
- Interpolation (n-D) using PreLookup：n 个输入信号的预插值；
- Lookup Table：输入信号的查询表（线性峰值匹配）；
- Lookup Table(2-D)：两维输入信号的查询表（线性峰值匹配）；
- Lookup Table(n-D)：n 维输入信号的查询表（线性峰值匹配）；
- Lookup Table Dynamic：动态查询表；
- PreLookup Index Search：预查询索引搜索；
- Sine：正弦函数查询表。

6．数学模块（Math Operations）

在 Simulink 基本模块中选择“Math Operations”后，单击便看到如图 3.20 所示的数学模块，它包括以下子模块。





图 3.20　数学模块

- Abs：取绝对值；
- Add：加法；
- Algebraic Constraint：代数约束；
- Assignment：赋值；
- Bias：偏移；
- Complex to Magnitude-Angle：由复数输入转为幅值和相角输出；
- Complex to Real-Imag：由复数输入转为实部和虚部输出；
- Divide：除法；
- Dot Product：点乘运算；
- Gain：比例运算；
- Magnitude-Angle to Complex：由幅值和相角输入合成复数输出；
- Math Function：包括指数函数、对数函数、求平方、开根号等常用数学函数；
- Matrix Concatenation：矩阵级联；
- MinMax：最值运算；
- MinMax Running Resettable：最大最小值运算；
- Polynomial：多项式；
- Product：乘运算；
- Product of Elements:：元素乘运算；
- Real-Imag to Complex：由实部和虚部输入合成复数输出；
- Reshape：取整；
- Rounding Function：舍入函数；
- Sign：符号函数；
- Sine Wave Function：正弦波函数；
- Slider Gain：滑动增益；
- Snbtract：减法；
- Sum：求和运算；
- Sum of Elements：元素和运算；
- Trigonometric Function：三角函数，包括正弦、余弦、正切等；
- Unary Minus：一元减法；
- Weighted Sample Time Math：权值采样时间运算。

7. 模型检测模块（Model Verification）

在 Simulink 基本模块中选择“Model Verification”后，单击便看到如图 3.21 所示的模型检测模块，它包模以下子模块。

- Assertion：确定操作；
- Check Discrete Gradient：检查离散梯度；
- Check Dynamic Gap：检查动态偏差；
- Check Dynamic Lower Bound：检查动态下限；

- Check Dynamic Range：检查动态范围；
- Check Dynamic Upper Bound：检查动态上限；
- Check Input Resolution：检查输入精度；
- Check Static Gap：检查静态偏差；
- Check Static Lower Bound：：检查静态下限；
- Check Static Range：检查静态范围；
- Check Static Upper Bound：检查静态上限。

8. 模型扩充模块（Model-Wide Utilities）

在 Simulink 基本模块中选择“Model-Wide Utilities”后，单击便看到如图 3.22 所示的模型扩充模块，它包括以下子模块。



图 3.21　模型检测模块

图 3.22　模型扩充模块

- Block Support Table：功能块支持的表；
- DocBlock：文档模块；
- Model Info：模型信息；
- Time-Based Linearization：时间线性分析；
- Trigger-Based Linearization：触发线性分析。

9. 端口和子系统模块（Port & Subsystems）

在 Simulink 基本模块中选择“Port & Subsystems”后，单击便看到如图 3.23 所示的端口和子系统模块，它包括以下子模块。

- Configurable Subsystem：结构子系统；
- Atomic Subsystem：单元子系统；

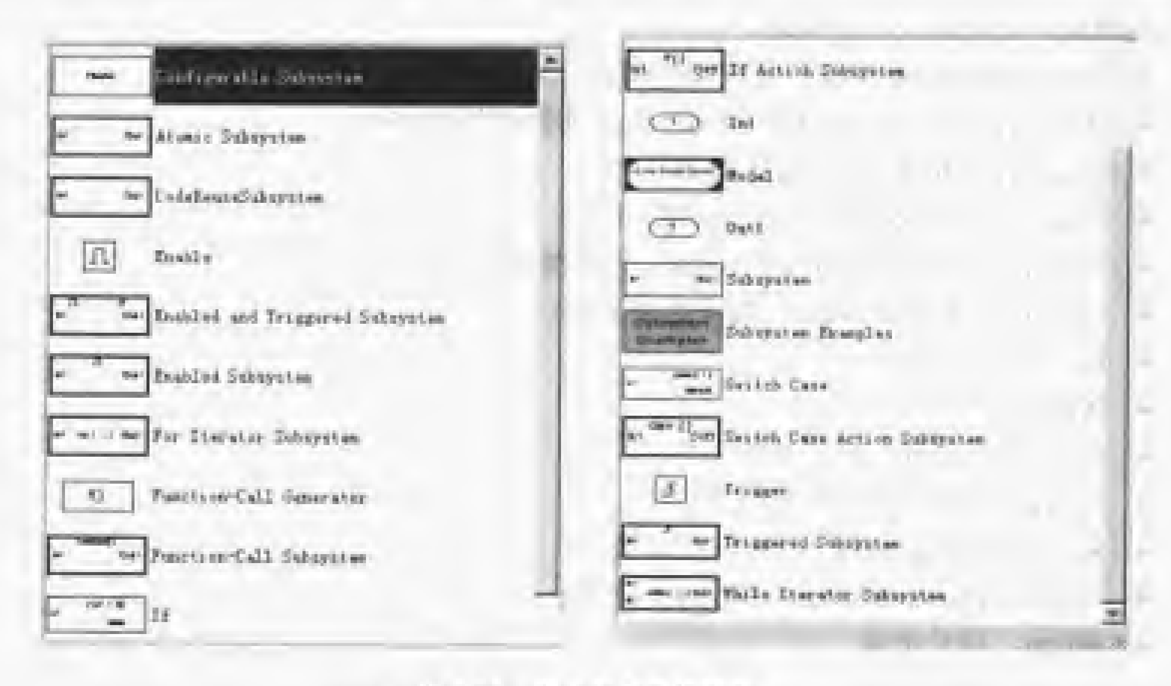


图 3.23　端口和子系统模块

- CodeReuse Subsystem：代码重用子系统；
- Enable：使能；
- Enable and Triggered Subsystem：使能和触发子系统；
- Enable Subsystem：使能子系统；
- For Iterator Subsystem：重复操作子系统；
- Function-Call Generator：函数响应生成器；
- Function-Call Subsystem 函数响应子系统；
- If：假设操作；
- If Action Subsystem：假设动作子系统；
- In1：输入端口；
- Model：模型；
- Out1：输出端口；
- Subsystem：子系统；
- Subsystem Examples：子系统例子；
- Switch Case：转换事件；
- Switch Case Action Subsystem：转换事件子系统；
- Trigger：触发操作；
- Triggered Subsystem：触发子系统；
- While Iterator Subsystem：重复子系统。

10．信号属性模块（Signal Attributes）

在 Simulink 基本模块中选择“Signal Attributes”后，单击便看到如图 3.24 所示的信

号属性模块，它包括以下子模块。

- Data Type Conversion：数据类型转换；
- Data Type Conversion Inherited：继承的数据类型转换；
- Data Type Duplicate：数据类型复制；
- Data Type Propagation：数据类型继承；
- Data Type Propagation Examples：数据类型继承例子；
- Data Type Scaling Strip：数据类型缩放；
- Ic：信号输入属性；
- Probe：探针点；
- Rate Transition：比率变换；
- Signal Conversion：信号转换；
- Signal Specification：信号特征说明；
- Weighted Sample Time：权值采样时间；
- Width：信号宽度



图 3.24　信号属性模块

11. 信号路线模块（Signal Routing）

在 Simulink 基本模块中选择“Signal Routing”后，单击便看到如图 3.25 所示的信号路线模块，它包括以下子模块。

- Bus Assignment：总线分配；
- Bus Creator：总线生成；
- Bus Selector：总线选择；



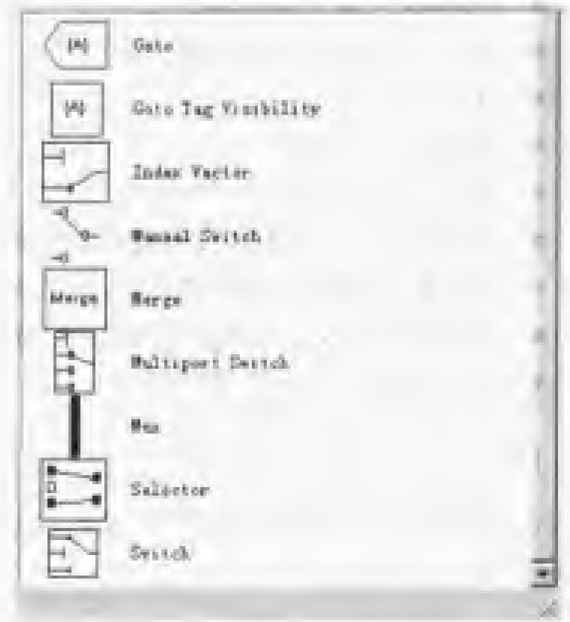


图 3.25　信号路线模块

- Data Store Memory：数据存储；
- Data Store Read：数据存储读取；
- Data Store Write：数据存储写入；
- Demux：将一个复合输入转化为多个单一输出；
- Environment Controller：环境控制器；
- From：信号来源；
- Goto：信号去向；
- Goto Tag Visibility：标签可视化；
- Index Vector：索引向量；
- Manual Switch：手动选择开关；
- Merge：信号合并；
- Multiport Switch：多端口开关；
- Mux：将多个单一输入转化为一个复合输出；
- Selector：信号选择器；
- Switch：开关选择，当第二个输入端大于临界值时，输出由第一个输入端而来，否则输出由第三个输入端而来。

12．接收器模块（Sinks）

在 Simulink 基本模块中选择“Sinks”后，单击便看到如图 3.26 所示的接收器模块，它包括以下子模块：



图 3.26　接收器模块

- Display：数字显示器；
- Floating Scope：浮动观察器；
- Out1：输出端口；
- Scope：示波器；
- Stop Simulation：仿真停止；
- Terminator：连接到没有连接到的输出端；
- To File(.mat)：将输出数据写入数据文件保存；
- To Workspace：将输出数据写入 MATLAB 的工作空间；
- XY Graph：显示二维图形。

13. 输入源模块（Sources）

在 Simulink 基本模块中选择"Sources"后，单击便看到如图 3.27 所示的输入源模块，它包括以下子模块。

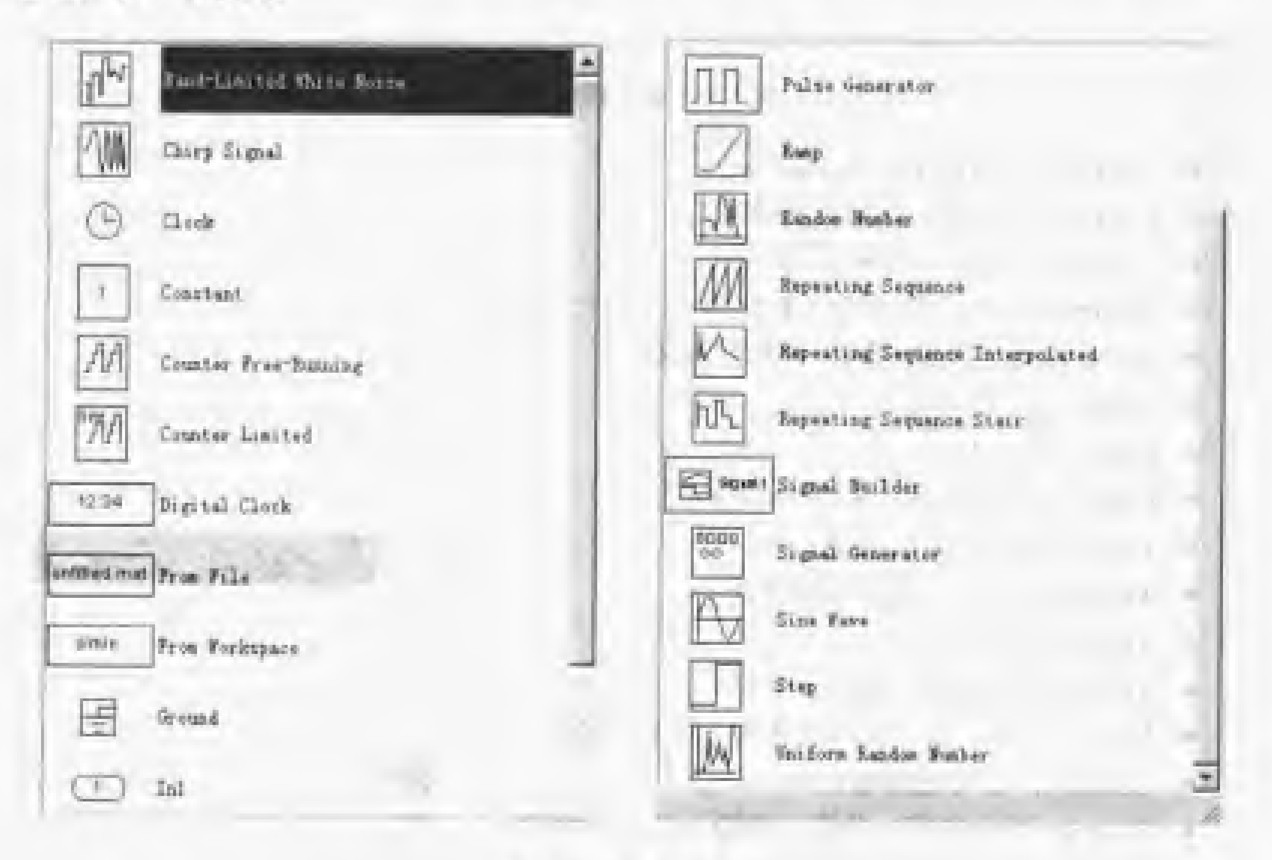


图 3.27　输入源模块

- Band-Limited White Noise：带限白噪声；
- Chirp Signal：产生一个频率不断增大的正弦波；
- Clock：显示和提供仿真时间；
- Constant：常数信号；
- Counter Free-Running：无限计数器；

- Counter Limited：有限计数器；
- Digital Clock：在规定的采样间隔产生仿真时间；
- From File(.mat)：来自数据文件；
- From Workspace：来自 MATLAB 的工作空间；
- Ground：连接到没有连接到的输入端；
- In1：输入信号；
- Pulse Generator：脉冲发生器；
- Ramp：斜坡输入；
- Random Number：产生正态分布的随机数；
- Repeating Sequence：产生规律重复的任意信号；
- Repeating Sequence Interpolated：重复序列内插值；
- Repeating Sequence Stair：重复阶梯序列；
- Signal Builder：信号创建器；
- Signal Generator：信号发生器，可以产生正弦、方波、锯齿波及随意波；
- Sine Wave：正弦波信号；
- Step：阶跃信号；
- Uniform Random Number：一致随机数；

14. 用户自定义函数模块（User-Defined Functions）

在 Simulink 基本模块中选择“User-Defined Functions”后，单击便看到如图 3.28 所示的用户自定义函数模块，它包括以下子模块。



图 3.28　用户自定义函数模块

- Embedded MATLAB Function：嵌入的 MATLAB 函数；
- Fcn：用自定义的函数（表达式）进行运算；
- M-file S-Function：M 文件编写的 S 函数；
- MATLAB Fcn：利用 MATLAB 的现有函数进行运算；
- S-Function：调用自编的 S 函数的程序进行运算；
- S-Function Builder：S 函数建立器；
- S-Function Examples：S 函数例子。

3.2.4　Simulink 功能模块的处理

3.2.4.1　Simulink 模块参数设置

1. 功能模块参数设置

在设置功能模块参数后，才能进行仿真操作。不同功能模块的参数是不同的，用鼠标双击该功能模块自动弹出相应的参数设置对话框。图 3.29 是传输延迟功能模块的对话框。

功能对话框由功能模块说明框和参数设置框组成。功能模块说明框用于说明该功能模块使用方法和功能；参数设置框用于设置该功能模块的参数。例如，传输延迟参数由最大延迟、初始输入、缓冲区的大小和 pade 近似的阶次组成，用户可输入相关参数。每个对话框的下面有“OK”（确认）、“Cancel”（取消）、“Help”（帮助）和“Apply”（应用）4 个按钮，设置功能模块参数后，需单击“OK”按钮进行确认，将设置参数送到仿真操作画面，并关闭对话框。单击“Cancel”按钮将取消刚才输入的设置参数，并关闭对话框。单击“Help”按钮，将弹出 Web 求助画面。单击“Apply”按钮将设置参数送仿真操作画面，但不关闭参数设置对话框。

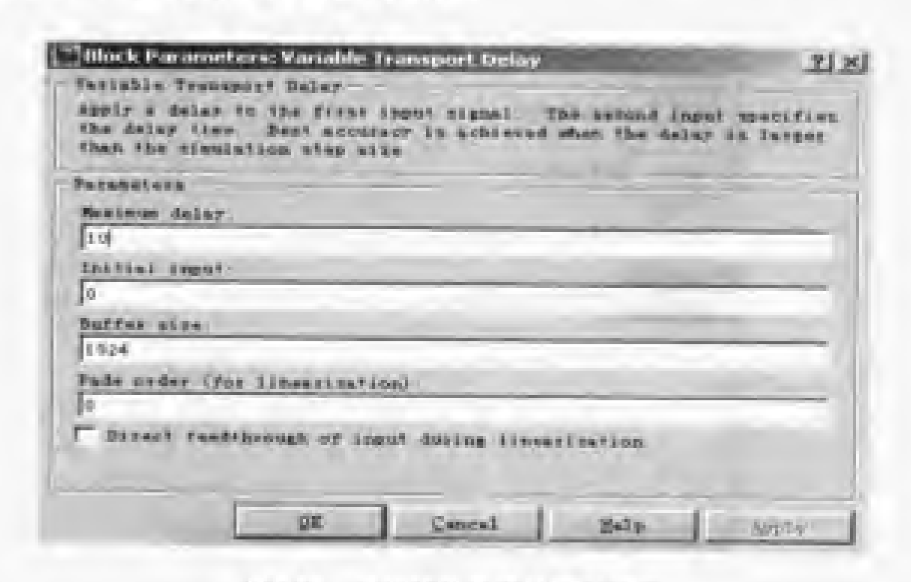


图 3.29　功能模块参数设置对话框

2. 示波器参数设置

采用 Simulink 仿真时，仿真结果通常通过示波器显示出来，示波器显示的结果直观且方便。需要对示波器的相关参数进行设置，示波器包括单踪和双踪示波器两种，下面分别对这两种示波器参数设置进行介绍。

（1）单踪示波器参数设置：单踪示波器显示的是输入信号与时间关系的曲线。

设置方法：双击已建立的示波器，将看到示波器显示界面，单击此对话框中工具条中的设置图标，将弹出如图 3.30 所示的示波器属性对话框。示波器属性对话框中包括两个标签，可用于设置坐标窗口数、显示时间范围、标记和显示频度或采样时间等。

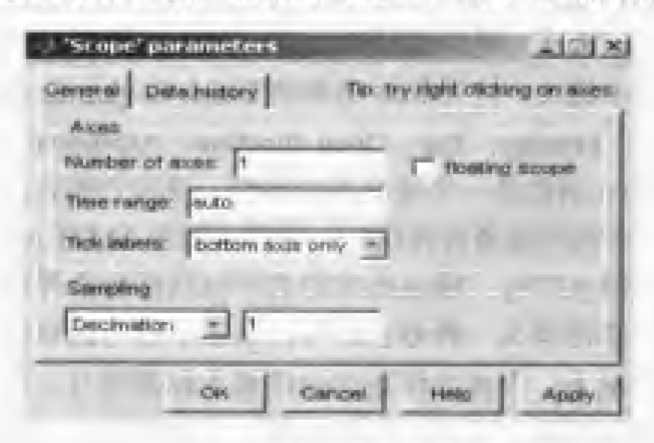


图 3.30　示波器属性对话框

（2）双踪示波器参数设置：双踪示波器显示的是两个输入信号之间关系的曲线。

设置方法同单踪示波器，示波器属性对话框中包括两个标签，可用于设置 x 坐标和 y 坐标的显示时间范围、标记和显示频度或采样时间等。

3.2.4.2　Simulink 模块基本操作

功能模块的基本操作包括模块的移动、复制、删除、转向、改变大小、模块命名、颜色设定、参数设定、属性设定、模块输入/输出信号等。模块库中的模块可以直接用鼠标进行拖曳（选中模块，按住鼠标左键不放）而放到模型窗口中进行处理。在模型窗口中，选中模块，则其 4 个角会出现黑色标记，此时可以对模块进行下述各操作。

（1）移动：选中模块，按住鼠标左键将其拖曳到所需的位置即可。若要脱离线而移动，可按住 Shift 键再进行拖曳。

（2）复制：选中模块，按住鼠标右键进行拖曳即可复制同样的一个功能模块。

（3）删除：选中模块，按 Delete 键即可。若要删除多个模块，可以同时按住 Shift 键，再用鼠标选中多个模块，按 Delete 键即可；也可以用鼠标选取某区域，再按 Delete 键就可以把该区域中的所有模块和线等全部删除。

（4）转向：为了能够顺序连接功能模块的输入和输出端，功能模块有时需要转向。在菜单 Format 中选择 Flip Block 旋转 180°，选择 Rotate Block 顺时针旋转 90°；或者直接按 Ctrl+F 组合键执行 Flip Block，按 Ctrl+R 组合键执行 Rotate Block。

（5）改变大小：选中模块，对模块出现的 4 个黑色标记进行拖曳即可。

（6）模块命名：先用鼠标在需要更改的名称上单击一下，然后直接更改即可。名称在功能模块上的位置也可以变换 180°，可以用 Format 菜单中的 Flip Name 来实现，也可以直接通过鼠标进行拖曳。Hide Name 可以隐藏模块名称。

（7）颜色设定：Format 菜单中的 Foreground Color 可以改变模块的前景颜色，Background Color 可以改变模块的背景颜色，而模型窗口的颜色可以通过 Screen Color 来改变。

（8）参数设定：用鼠标双击模块就可以进入模块的参数设定窗口，从而对模块进行参数设定。参数设定窗口包含了该模块的基本功能帮助，为获得更详尽的帮助，可以单击其上的"Help"按钮。通过对模块的参数设定，就可以获得需要的功能模块。

（9）属性设定：选中模块，打开 Edit 菜单的 Block Properties 可以对模块进行属性设定，包括对 Description、Priority、Tag、Open functiou、Attributes format string 等属性的设定。其中 Open function 属性是一个很有用的属性，通过它指定一个函数名，当模块被双击之后，Simulink 就会调用该函数并执行，这种函数在 MATLAB 中称为回调函数。

（10）模块的输入/输出信号：模块处理的信号包括标量信号和向量信号。标量信号是一种单一信号，而向量信号为一种复合信号，是多个信号的集合，它对应着系统中几条连线的合成。默认情况下，大多数模块的输出都为标量信号，对于输入信号，模块都具有一种"智能"的识别功能，能自动进行匹配。某些模块通过对参数的设定，可以使模块输出向量信号。

3.2.4.3　Simulink 模块间连线处理

Simulink 模型的构建是通过用线将各种功能模块进行连接而构成的。用鼠标可以在功能模块的输入端与输出端之间直接连线。所画的线可以改变粗细、设定标签，也可以把线折弯、分支。

（1）改变粗细：线所以有粗细是因为线引出的信号可以是标量信号或向量信号，当选中 Format 菜单下的 Wide Vector Lines 时，线的粗细会根据线所引出的信号是标量还是向量而改变，如果信号为标量则为细线，若为向量则为粗线。选中 Vector Line Widths 则可以显示出向量引出线的宽度，即向量信号由多少个单一信号合成。

（2）设定标签：只要在线上双击鼠标，即可输入该线的说明标签。也可以通过选中线，然后打开 Edit 菜单下的 Signal Properties 进行设定，其中 Signal name 属性的作用是标明信号的名称，设置这个名称反映在模型上的直接效果就是与该信号有关的端口相连的所有直线附近都会出现写有信号名称的标签。

（3）线的折弯：按住 Shift 键，再用鼠标在要折弯的线处单击一下，就会出现圆圈，表示折点，利用折点就可以改变线的形状。

（4）线的分支：按住鼠标右键，在需要分支的地方拉出即可，或者按住 Ctrl 键并在要建立分支的地方用鼠标拉出即可。

3.3 Simulink 自定义功能模块

自定义功能模块有两种方法，一种方法是采用 Signal&Systems 模块库中的 Subsystem 功能模块，利用其编辑区设计组合新的功能模块；另一种方法是将现有的多个功能模块组合起来，形成新的功能模块。对于很大的 Simulink 模型，通过自定义功能模块可以简化图形，减少功能模块的个数，有利于模型的分层构建。

3.3.1 采用 Subsystem 构建自定义功能模块

将 Signal&Systems 模块库中的 Subsystem 功能模块复制到打开的模型窗口中。

双击 Subsystem 功能模块，进入自定义功能模块窗口，即可利用已有的基本功能模块设计出新的功能模块。

3.3.2 多个模块组合自定义功能模块

在模型窗口中建立所定义功能模块的子模块。

用鼠标将这些需要组合的功能模块选中，然后选择 Edit 菜单下的 Create Subsystem 即可。

3.3.3 自定义功能模块的封装

上面提到的两种方法都只是创建一个功能模块而已，如果要命名该自定义功能模块、对功能模块进行说明、选定模块外观、设定输入数据窗口，则需要对其进行封装处理。

首先选中 Subsystem 功能模块，再打开 Edit 菜单中的 Mask Subsystem 进入 mask 的编辑窗口，可以看出有 3 个标签页。

1. Icon 标签页

它用于设定功能模块外观，最重要的部分是 Drawing Commands，在该区域内可以用 disp 指令设定功能模块的文字名称，用 plot 指令画线，用 dpoly 指令画转换函数。

注意：尽管这些命令在名字上和以前讲的 MATLAB 函数相同，但它们在功能上却不完全相同，因此不能随便套用以前所讲的格式。

- disp('text')：在功能模块上显示设定的文字内容。
- disp('text1\ntext2')：分行显示文字 text1 和 text2
- plot([x1 x2 … xn], [y1 y2 … yn])：在功能模块上画出由[x1 y1]经[x2 y2]经[x3 y3]…直到[xn, yn]为止的直线。功能模块的左下角会根据目前的坐标刻度被正规化为[0, 0]，右上角则会依据目前的坐标刻度被正规化为[1, 1]。
- dpoly(num, den)：按 s 次数的降幂排序，在功能模块上显示连续的传递函数。
- dpoly(num, den, 'z')：按 z 次数的降幂排序，在功能模块上显示离散的传递函数。

用户还可以设置一些参数来控制图标的属性，这些属性在 Icou 页右下端的下拉式列表中进行选择。

- Icon frame：选择 Visible 则显示外框线；选择 Invisible 则隐藏外框线。
- Icon Transparency：选择 Opaque 则隐藏输入/输出的标签；选择 Transparent 则显示输入/输出的标签。
- Icon Rotation：旋转模块。
- Drawing coordinate：画图时的坐标系。

2. Initialization 标签页

它用于设定输入数据窗口（Prompt List），它主要用来设计输入提示（prompt）以及对应的变量名称（variable）。在 prompt 栏上输入变量的含义，其内容会显示在输入提示中。variable 是仿真要用到的变量，该变量的值一直存于 mask workspace 中，因此可以与其他程序相互传递。

如果配合在 initialization commands 内编辑程序，则可以发挥功能模块的功能来执行特定的操作。

（1）在 prompt 编辑框中输入文字，这些文字就会出现在 prompt 列表中；在 variable 列表中输入变量名称，则 prompt 中的文字对应该变量的说明。如果要增加新的项目，则可以单击边上的 Add 键。Up 和 Down 按钮用于执行项目间的位置调整。

（2）Control type 列表给用户提供选择设计的编辑区，选择 Edit 会出现供输入的空白区域，所输入的值代表对应的 variable；Popup 则为用户提供可选择的列表框，所选的值代表 variable，此时在下面会出现 Popup strings 输入框，用来设计选择的内容，各值之间用逻辑或符号“|”隔开；若选择 Checkbox 则用于 on 与 off 的选择设定。

（3）Assignment 属性用于配合 Control type 的不同选择来提供不同的变量值，变量值分为 Evaluate 和 Literal 两种，其含义如表 3.1 所示。

表 3.1　Assignment 属性的含义

Control Type	Assignment	
	Evaluate	Literal
Edit	输入的文字是程序执行时所用的变量值	输入内容做字符串处理
Popup	所选序号，选第一项输出 1，以此类推	选择内容做字符串处理
Checkbox	输出为 1 或 0	输出为‘on’或‘off’的字符串

3. Documentatiou 标签页

它用于设计该功能模块的文字说明，主要针对完成的功能模块来编写相应的说明文字和 Help。

（1）在 Block description 中输入的文字，会出现在参数窗口的说明部分。

（2）在 Block help 中输入的文字会显示在单击参数窗口中的“Help”按钮后浏览器所加载的 HTML 文件中。

（3）在 Mask type 中输入的文字作为封装模块的标注性说明，在模型窗口下，将鼠标指向模块则会显示该文字。当然必须先在 View 菜单中选择 Block Data Tips——Show

Block Data Tips。

3.4 S 函数设计与应用

Simulink 为用户提供了许多内置的基本库模块，如连续系统模块库（Continous）、离散系统模块库（Discontinous）等，通过这些模块的连接构成系统的模型。这些内置的基本库模块是有限的，在许多情况下，尤其是在特殊的应用中，需要用到一些特殊的模块，这些模块可以用基本模块构成，是由基本模块扩展而来的。

Simulink 提供了一个功能强大的对模块库进行扩展的新工具 S-Function，它依然是基于 Simulink 原来提供的内置模块，通过对那些经常使用的模块进行组合并封装而构建出可重复使用的新模块。

S-Function 是系统函数（System Function）的简称，是一个动态系统的计算机语言描述。在 MATLAB 中，用户可以选择用 M 文件编写，也可以用 C 或 mex 文件编写，在这里只给大家介绍如何用 M 文件编写 S-Function，使用 C 语言或 mex 文件编写的方法与 M 文件编写的方法基本类似。

S-Function 提供了扩展 Simulink 模块库的有力工具，它采用一种特定的调用语法，实现函数和 Simulink 解法器之间的交互。

S-Function 最广泛的用途是定制用户自己的 Simulink 模块。它的形式十分通用，能够支持连续系统、离散系统和混合系统。

3.4.1 S 函数设计

对于一些算法比较复杂的模块可以使用 MATLAB 语言按照 S-Function 的特式来编写。应该注意的是，这样构造的 S-Function 只能用于基于 Simulink 的仿真，并不能将其转换成独立于 MATLAB 的程序。

MATLAB 提供了一个模板文件，方便了 S-Fnnction 的编写，该模板文件位于 MATLAB 根目录 toolbox/Simulink/blocks 下。去除注释部分后的程序如下所示：

```
S-Function 的模板
function [sys, x0, str, ts]  =  sfuntmpl(t, x, u, flag)
switch flag,
case 0,
% Initialization
[sys, x0, str, ts]=mdlInitializeSizes;
case 1,
% Derivatives
sys=mdlDerivatives(t, x, u);
case 2,
% Update
sys=mdlUpdate(t, x, u);
```

```
case 3,
% Outputs
sys=mdlOutputs(t, x, u);
case 4,
% GetTimeOfNextVarHit
sys=mdlGetTimeOfNextVarHit(t, x, u);
case 9,
% Terminate
sys=mdlTerminate(t, x, u);
otherwise
% Unexpected flags
error(['Unhandled flag = ', num2str(flag)]);
end
```

其中的子函数如下所示：

```
S-Function 的子函数
function [sys, x0, str, ts]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 0;
sizes.NumInputs      = 0,
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];
function sys=mdlDerivatives(t, x, u)
sys = [];
function sys=mdlUpdate(t, x, n).
sys = [];
function sys=mdlOutputs(t, x, n)
sys = [];
function sys=mdlGetTimeOfNextVarHit(t, x, u)
sampleTime = 1;
sys = t + sampleTime;
function sys=mdlTerminate(t, x, u)
sys = [];
```

模板文件中 S-Functiou 的结构十分简单，它只为不同的 flag 值指定需调用的 M 文件

子函数。例如，当 flag＝3 时，即模块处于计算输出这个仿真阶段时，需调用的子函数为 sys＝mdloutputs(t, x, u)。

模板文件使用 switch 语句来完成这种指定，当然这种结构并不惟一，用户也可以使用 if 语句来完成同样的功能。在实际运用时，可以根据实际需要去掉某些值，因为并不是每个模块都需要经过所有的子函数调用。

模板文件只是 Simulink 为方便用户而提供的一种参考格式，并不是编写 S-Function 的语法要求，用户完全可以改变子函数的名称，或者直接把代码写在主函数中。但使用模板文件的好处是比较方便、条理清晰。

使用模板编写 S-Function，用户只需把 S 函数名换成期望的函数名称，如果需要额外的输入参量，还需在输入参数列表的后面增加这些参数，因为前面的 4 个参数是 Simulink 调用 S-Function 时自动传入的。对于输出参数，最好不修改。接下来的工作就根据所编 S-Function 要完成的任务，用相应的代码去替代模板里各个子函数的代码。

Simulink 在每个仿真阶段都会对 S-Function 进行调用，在调用时，Simulink 会根据所处的仿真阶段为 flag 传入不同的值，而且还会为 sys 这个返回参数指定不同的角色，即尽管是相同的 sys 变量，但在不同的仿真阶段其意义却不相同，这种变化由 Simulink 自动完成。

M 文件 S-Function 可用的子函数说明如下。

- mdlInitializeSizes：定义 S-Function 模块的基本特性，包括采样时间、连续或者离散状态的初始条件和 Sizes 数组。
- mdlDerivatives：计算连续状态变量的微分方程。
- mdlUpdate：更新离散状态、采样时间和主时间步的要求。
- mdlOutputs：计算 S-Function 的输出。
- mdlGetTimeOfNextVarHit：计算下一个采样点的绝对时间，即在 mdlInitializeSizes 中说明了一个可变的离散采样时间。
- mdlTerminate：结束仿真任务。

概括说来，建立 S-Function 可以分成两个分离的任务：

- 初始化模块特性包括输入/输出信号的宽度、离散连续状态的初始条件和采样时间。
- 将算法放到合适的 S-Function 子函数中去。

为了让 Simulink 识别出一个 M 文件 S-Function，用户必须在 S 函数里提供有关 S 函数的说明信息，包括采样时间、连续或者离散状态个数等初始条件。这一部分主要是在 mdlInitializeSizes 子函数里完成。

Sizes 数组是 S-Function 函数信息的载体，其内部的字段意义分别如下所述。

- NumContStates：连续状态的个数（状态向量连续部分的宽度）。
- NumDiscStates：离散状态的个数（状态向量离散部分的宽度）。
- NumOutputs：输出变量的个数（输出向量的宽度）。
- NumInputs：输入变量的个数（输入向量的宽度）。
- DirFeedthrough：有无直接馈入。注意：DirFeedthrough 是一个布尔变量，它的取

值只有 0 和 1 两种。0 表示没有直接馈入，此时用户在编写 mdlOutputs 子函数时就要确保子函数的代码里不出现输入变量 u；1 表示有直接馈入。

- NumSampleTimes：采样时间的个数，也就是 ts 变量的行数，与用户对 ts 的定义有关。

如果字段代表的向量宽度为动态可变，则可以将它们赋值为–1。需要指出的是，由于 S-Function 会忽略端口，所以当有多个输入变量或多个输出变量时，必须用 mux 模块或 demux 模块将多个单一输入合成为一个复合输入向量或将一个复合输出向量分解为多个单一输出。

S-Function 默认的 4 个输入参数为 t、x、u 和 flag，它们的次序不能变动，代表的意义分别为：

- t 表示当前仿真时间，这个输入参数通常用于决定下一个采样时刻，或者在多采样速率系统中，用来区分不同的采样时刻点，并据此进行不同的处理。
- x 表示状态向量，这个参数是必须的，甚至在系统中不存在状态时也是如此，它的使用非常灵活。
- u 表示输入向量。
- flag 是一个用于控制在每一个仿真阶段调用哪一个子函数的参数，由 Simulink 在调用时自动取值。

S-Function 默认的 4 个返回参数为 sys、x0、str 和 ts，它们的次序不能变动，代表的意义分别为：

- sys 是一个通用的返回参数，其返回值的意义取决于 flag 的值。
- x0 是初始的状态值（没有状态时是一个空矩阵），这个返回参数只在 flag 值为 0 时才有效，其他时候都会被忽略。
- str 参数没有什么意义，是 MathWorks 公司为将来的应用保留的，M 文件 S-Function 必须把它设为空矩阵。
- ts 是一个 $m\times2$ 矩阵，它的两列分别表示采样时间间隔和偏移。

3.4.2 S 函数应用

下面用一个例子说明 S 函数的应用方法。

【例 3-1】 利用 MATLAB 中 S 函数模板设计一个离散系统的 S-Function。

解：MATLAB 程序代码如下。

```
function [sys, x0, str, ts] = dsfunc(t, x, u, flag)
% 定义一个离散系统的 S-function
%       x(n+1) = Ax(n) + Bu(n)
%       y(n) = Cx(n) + Du(n)
A=[-1.3839, -0.5097; 1.0000, 0];
B=[-2.5559, 0; 0, 4.2382];
C=[0, 2.0761; 0, 7.7891];
D=[-0.8141, -2.9334; 1.2426, 0];
switch flag,
```

```
    case 0,
      [sys, x0, str, ts] = mdlInitializeSizes(A, B, C, D);
    case 2,
      sys = mdlUpdate(t, x, u, A, B, C, D);
    case 3,
      sys = mdlOutputs(t, x, u, A, C, D);
    case 9,
      sys = []; % do nothing
    otherwise
      error(['unhandled flag  —  ', num2str(flag)]),
end
function [sys, x0, str, ts] = mdlInitializeSizes(A, B, C, D)
sizes = simsizes;
sizes.NumContStates = 0;
%两个离散状态
sizes.NumDiscStates = size(A, 1);
%两个输出
sizes.NumOutputs = size(O, 1),
%两个输入
sizes.NumInputs = size(D, 2);
sizes.DirFeedthrongh = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
%初始状态为 1
x0 = ones(sizes.NumDiscStates, 1);
str = [];
%时间间隔为 1，时间偏移为 0
ts = [1 0];
function sys = mdlUpdate(t, x, u, A, B, C, D)
sys = A*x+B*u;
sys = C*x+D*u;
```

3.5　Simulink 仿真举例

采用 Simulink 进行仿真，不仅系统模型的搭建简单方便，而且能直接获得系统输出或状态变量变化曲线，具有简单明了、直观形象的特点，得到了广泛应用。

【例 3-2】　已知一闭环系统结构如图 3.31 所示，其中，系统前向通道的传递函数为 $G(s)=\dfrac{s+0.5}{s+0.1}\cdot\dfrac{20}{s^3+12s^2+20s}$，而且前向通道有一个[-0.2, 0.5]的限幅环节，图中用 N

表示，反馈通道的增益为 1.5，系统为负反馈，阶跃输入经 1.5 倍的增益作用到系统，试利用 Simulink 对该闭环系统进行仿真，要求观测其单位阶跃响应曲线。

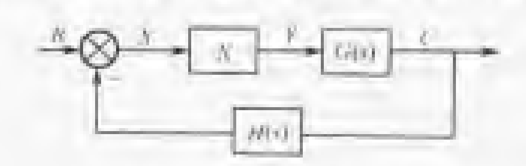


图 3.31　例 3-2 的系统框图

解：使用 Simulink 进行仿真的基本步骤如下。

（1）在 MATLAB7.0 的窗中双击 Simulink 图标就打开 Simulink Library Browser 窗口，在此窗口进入 File\New\Model，就会打开一个 untitled 窗口（可以用 "Save as" 保存此窗口并改名），如图 3.32 所示。

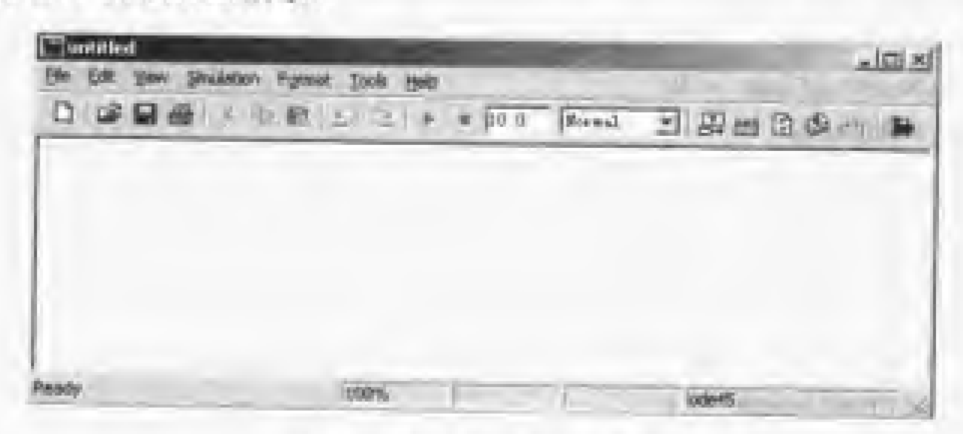


图 3.32　控制模型框图编辑窗口

（2）根据题意在 Simulink Library Browser 中选定需要使用的子模块，如图 3.33 所示。在本例中，需要单位阶跃信号、增益模块、表示连续系统的模块、表示限幅环节的模块、用来把输入信号和输出信号组合起来以便直观观察的模块、把输入信号和反馈信号综合的模块、把仿真中的变量输入到工作空间的模块、观测系统响应曲线的模块和时钟模块。在 Simulink Library Browser 窗口下找到了符合要求的模块：表示阶跃信号的模块 Step，它位于 SOURCE 模块组中；表示增益的模块 Gain，它位于 MATH OPERATIONS 模块组中；表示连续系统的模块 Transfer Fcn，它位于 CONTINOUS 模块组中；表示限幅环节的模块 Saturation，它位于 DISCONTINUTIES 模块组中；用来把输入信号和输出信号组合的模块 Mux，它位于 Signal Routing 模块组中；用于把输入信号和反馈信号综合的模块 Sum，它位于 MATH OPERATIONS 模块组中；用于把仿



图 3.33　Simulink Library Browser 窗口

真变量输入到工作空间的模块 To Workspace，它位于 SINKS 模块组中；用于观测系统响应曲线的模块 Scope，它位于 Sinks 模块组中；用来产生时钟信号的模块 Clock，它位于 SOURCE 模块组中。

把选定好的模块依次拖到 untitled 窗口中，如图 3.34 所示。注意：模块的选择有时不是惟一的，要根据自己的习惯定。在本例中，表示系统前向通道传递函数的模块可以用 Transfer Fcn，也可以用 Zero-Pole 或 State-Space，当然，需要进行简单的转换，详见第 4 章。

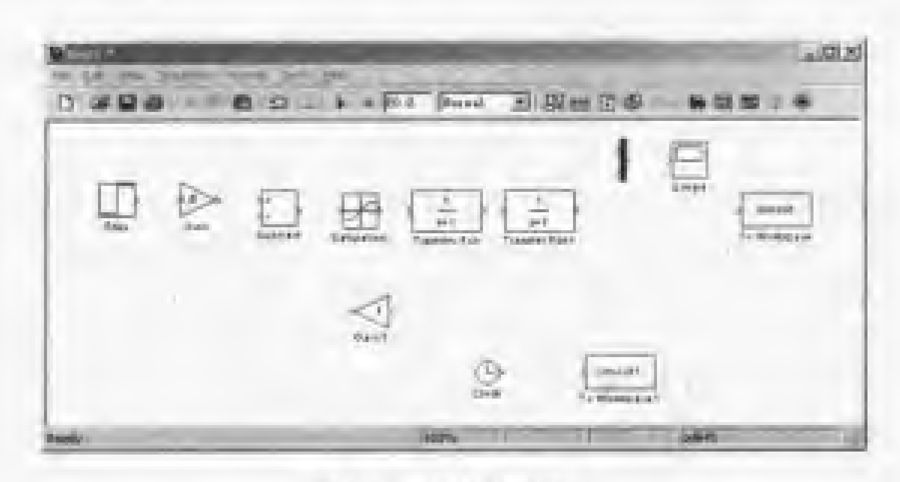


图 3.34　模块编辑窗口

（3）连接模块并设定模块参数。把各功能模块按照逻辑关系连接起来，双击某一个模块，就会出现该模块的设置窗口，如图 3.35 所示，依次设置模块的参数。以图 3.35 为例，它是传递函数的设置窗口，设定它为 $\dfrac{s+0.5}{s+0.1}$，则在 Numerator 中输入[1 0.5]，在 Denominator 中输入[1 0.1]，其他的选项按默认值设定，然后单击“OK”按钮完成设置。

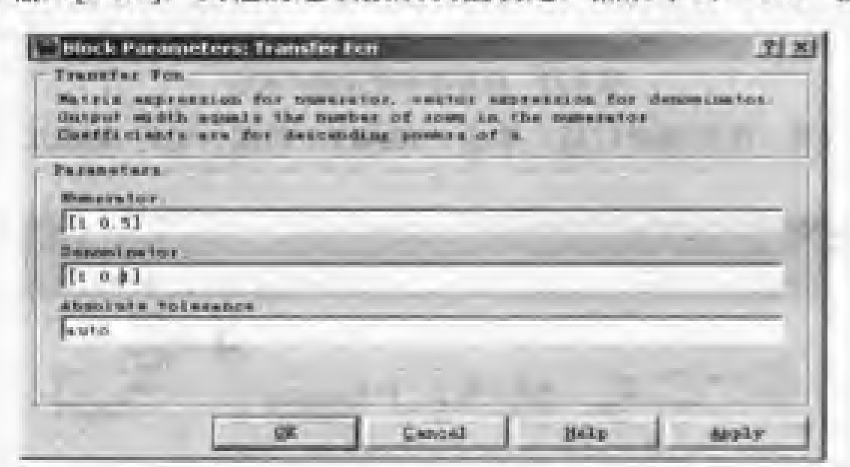


图 3.35　模块参数设定窗口

在 Simulink 仿真中常用的一个模块是 To Workspace，它把 Simulink 仿真的数据传送到工作空间中，以供 MATLAB 程序使用。用鼠标双击“To Workspace”图标，得到如图 3.36 所示的 To Workspace 模块参数对话框。

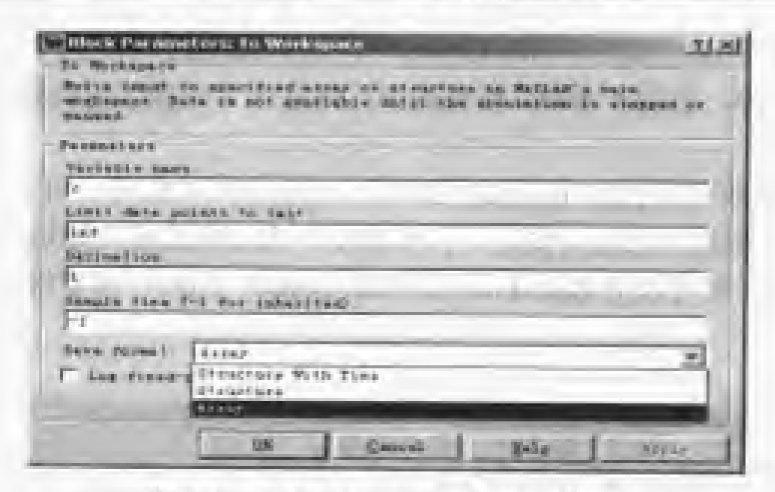


图 3.36　To Workspace 模块参数对话框

本例中，需要传输数据向量 c 和 t。以设置数据向量 c 为例，在 Variable name 编辑框中输入向量名 c，save format 编辑选择 Array（向量）项，然后单击“OK”按钮完成设置。仿真运行后，向量 c(t)和 t 以各自变量名存在于 MATLAB Workspace 中。

（4）设置仿真器参数，仿真器设置窗口如图 3.37 所示，在 SIMULATION\SIMULATION PARAMETERS\SOLVER 中设置 SolverType、Solver（步长）、Stop Time 等。

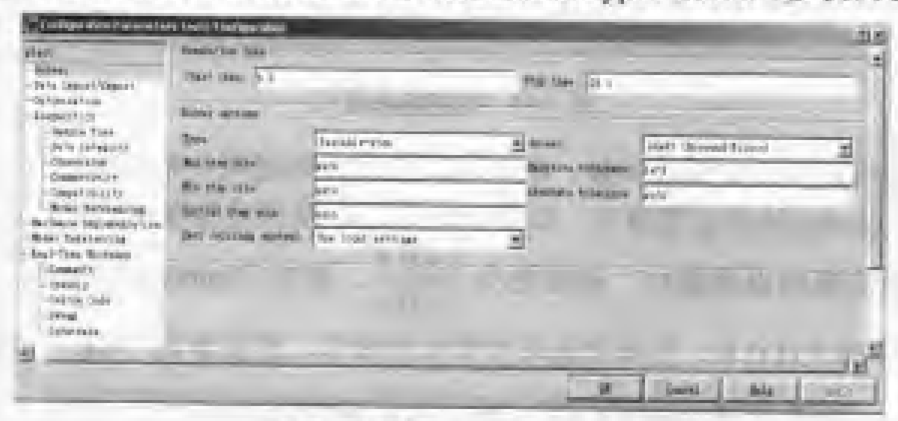


图 3.37　仿真器参数设置对话框

（5）运行仿真。模型编辑好后，单击“START”按钮，运行 SIMULATION\START。如图 3.38 所示。

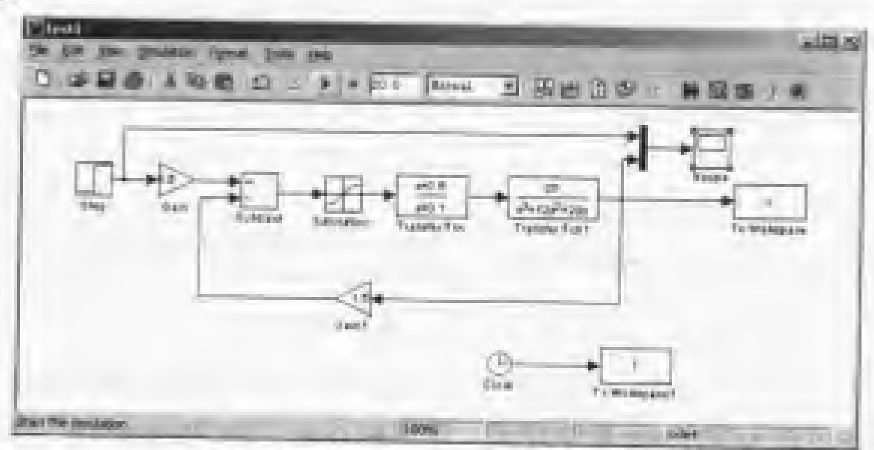


图 3.38　例 3-2 的 Simulink 仿真框图

（6）分析仿真结果。仿真中，一般采用示波器观测输出结果。双击 Scope 模块，输出的图形如图 3.39 所示。由于采用了 Mux 模块，把输入信号和输出信号合在一起同时显示，所以图上是两条曲线，它会自动分配不同的颜色以便观察。



图 3.39　示波器输出结果

（7）对工作空间中的数据做后续处理。当仿真任务比较复杂时，需要把 Simulink 生成的数再导入到工作空间中来进行一些处理和分析。在本例中，Simulink 仿真结束后，输出结果通过 To workspace 传送倒工作空间中，如图 3.40 所示。在工作空间窗口中能看到这些变量，使用“whos”命令能看到这些变量的详细信息。另外，还有一些模块（如 From file、To file）能实现文件与 Simulink 的数据传输，读者可以通过模块的使用说明和有关对话框的引导进行使用。

有关 Simulink 的使用将在后面有关自动控制系统分析和设计的章节中进一步详细论述。



图 3.40　工作空间窗口

第 4 章　控制系统数学模型

4.1　引　　言

控制系统的数学模型在控制系统的研究中有着相当重要的地位，要对系统进行仿真处理，首先需要知道系统的数学模型，而后才有可能对系统进行模拟。同样，只有知晓系统模型，才有可能在此基础上设计一个合适的控制器，使系统响应达到预期效果，满足实际的工程需要。

在线性系统理论中，常用的数学模型形式有：传递函数模型（系统的外部模型）、状态方程模型（系统的内部模型）、零极点增益模型和部分分式模型等。这些模型之间都有着内在的联系，可以相互进行转换。

本章介绍控制系统数学模型的建立、模型的分类、模型之间的转换以及如何利用 MATLAB 建模和对模型进行转换。通过本章，读者对控制系统数学模型的基本知识能有所了解，并学会利用 MATLAB 进行建模。

4.2　动态过程微分方程描述

微分方程是控制系统模型的基础，一般来讲，利用机械学、电学、力学等物理规律便可以得到控制系统的动态方程，这些方程对于线性定常连续系统而言是一种常系数的线性微分方程。

控制系统动态微分方程的建立基于以下两个条件：

（1）在给定量产生变化或扰动出现之前，被控制量的各阶导数都为零，即系统是处于平衡状态的，因此，在任一瞬间，由各种不同环节组成的自动控制系统用几个独立变量就可以完全确定系统的状态。

（2）建立的动态微分方程式是以微小增量为基础的增量方程，而不是其绝对值的方程，因此，当出现扰动或给定量产生变化时，被控量和各独立变量在其平衡点附近将产生微小的增量，微分方程式描述的是微小偏差下系统运动状态的增量方程，不是运动状态变量的绝对值方程，也不是大偏差范围内的增量方程。

动态微分方程描述的是被控制量与给定量或扰动量之间的函数关系，给定量和扰动量可以看成系统的输入量，被控制量看成输出量。建立微分方程时，一般从系统的环节着手，先确定各环节的输入量和输出量，以确定其工作状态，并建立各环节的微分方程，而后消去中间变量，最后得到系统的动态微分方程。

【例 4-1】　建立如图 4.1 所示的 LRC 电路的微分方程式。

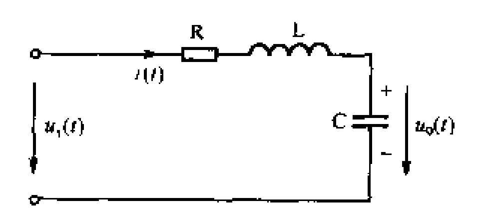


图 4.1 RLC 电路示意图

解：步骤 1：确定电路的输入和输出量。

由图可知，当电压 $u_i(t)$ 发生变化时，将引起电路中电流 $i(t)$ 的变化，因此 $u_o(t)$ 也将随着变化。在这里，取电路的输入量为 $u_i(t)$，以下式表示：

$$x_{\text{reason}} = u_i(t)$$

取输入量为 $u_o(t)$，以下式表示：

$$x_{\text{cause}} = u_o(t)$$

这两个量都是时间的函数，而回路中的电路 $i(t)$ 则为联系输入和输出的中间变量。

步骤 2：列出原始微分方程式。

根据基尔霍夫电压定律、电流定律得到系统的原始微分方程为：

$$L\frac{\mathrm{d}i(t)}{\mathrm{d}t} + R\cdot i(t) + u_o(t) = u_i(t)$$

$$i(t) = C\frac{\mathrm{d}u_o(t)}{\mathrm{d}t}$$

步骤 3：消去中间变量。

联立以上方程式，消去中间变量 $i(t)$，可得到电路微分方程式：

$$LC\frac{\mathrm{d}^2u_o(t)}{\mathrm{d}t^2} + RC\cdot\frac{\mathrm{d}u_o(t)}{\mathrm{d}t} + u_o(t) = u_i(t)$$

或

$$LC\frac{\mathrm{d}^2x_{\text{cause}}}{\mathrm{d}t^2} + RC\cdot\frac{\mathrm{d}x_{\text{cause}}}{\mathrm{d}t} + x_{\text{cause}} = x_{\text{reason}}$$

对于比较复杂的系统，建立系统微分方程一般可采用以下步骤：

（1）将系统划分为多个环节，确定各环节的输入及输出信号，每个环节可考虑写一个方程；

（2）根据物理定律或通过实验等方法得出物理规律，列出各环节的原始方程式，并考虑适当简化、线性化；

（3）将各环节方程式联立，消去中间变量，最后得出只含有输入变量、输出变量以及参量的系统方程式。

单输入单输出系统微分方程表示的输入模型一般具备如下形式：

$$\begin{aligned} &a_0x_o^{(n)}(t) + a_1x_o^{(n-1)}(t) + \cdots + a_{n-1}x_o^{(1)}(t) + a_nx_o(t) \\ &= b_0x_i^{(m)}(t) + b_1x_i^{(m-1)}(t) + \cdots + b_{m-1}x_i^{(1)}(t) + b_mx_i(t) \end{aligned} \tag{4-1}$$

如果已知输入量及变量的初始条件，对微分方程进行求解，就可以得到系统输出量

的表达式，并由此对系统进行性能分析。MATLAB 提供了 ode23、ode45 等微分方程的数值解法函数，不仅适用于线性定常系统，也适用于非线性和时变系统。

【例 4-2】 电路如图 4.2 所示，$R=1.6\ \Omega$，$L=2.1$ H，$C=0.30$ F，初始状态是电感电流为零，电容电压为 0.2V，$t=0$ 时接入 1.5V 的电压，求 $0<t<10$s 时 $i(t)$，$u_o(t)$的值，并画出电流与电容电压的关系曲线。

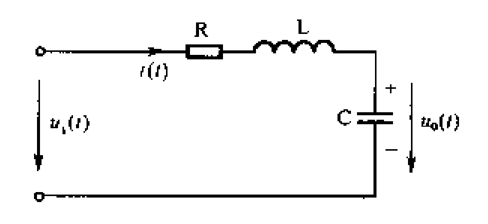


图 4.2　RLC 电路示意图

解：程序主函数的 MATLAB 代码如下。

```
clear
clc
close
t0=0;
tfinal=10;
%初始化，电感电流为 0，电容电压为 0.2v
x0=[0.2;0];
%rlcsys 是系统微分方程的描述函数
[t, x]=ode45('rlcsys', t0, tfinal, x0);
figure(1）
subplot(211）
plot(t, x(:, 1)）
%添加栅格
grid
title('电容电压/v'）
xlabel('时间/s'）
subplot(212）
plot(t, x(:, 2)）
%添加栅格
grid
title('电感电流/A'）
xlabel('时间/s'）
figure(2）
vc=x(:, 1);
i=x(:, 2);
plot(vc, i);
```

```
%添加栅格
grid
title('电感电流与电容电压的关系曲线')
xlabel('电容电压/v')
ylabel('电感电流/A')
```

系统微分方程描述函数的 MATLAB 如下：

```
%微分方程函数
%状态导数
function xdot=rlcsys(t, x)
%电压值
Vs=1.5;
%电阻值
R=1.6;
%电感值
L=2.1;
%电容值
C=0.30;
%导数关系式
xdot=[x(2)/C;1/L*(Vs-x(1)-R*x(2))];
%格式
%  function xdot=filename(t, x)
%  xdot=[表达式 1；表达式 2；表达式 3；...；表达式 n-1]
%  表达式 1 对应  x1'=x2
%  表达式 2 对应  x2'=x3
%  表达式 3 对应  x3'=x4
%  ...
%  表达式 n-1 对应  xn-1'=xn
%  本例中, x(1)=Vo, x(2)=iL, x(1)'=x(2)/C
%  x(2)'=(Vs-x(1)-R*x(2))/L
```

程序运行结果如图 4.3 和图 4.4 所示，图 4.3 为 $i(t)$，$u_o(t)$的时间关系曲线，图 4.4 为电流与电容电压的关系曲线。

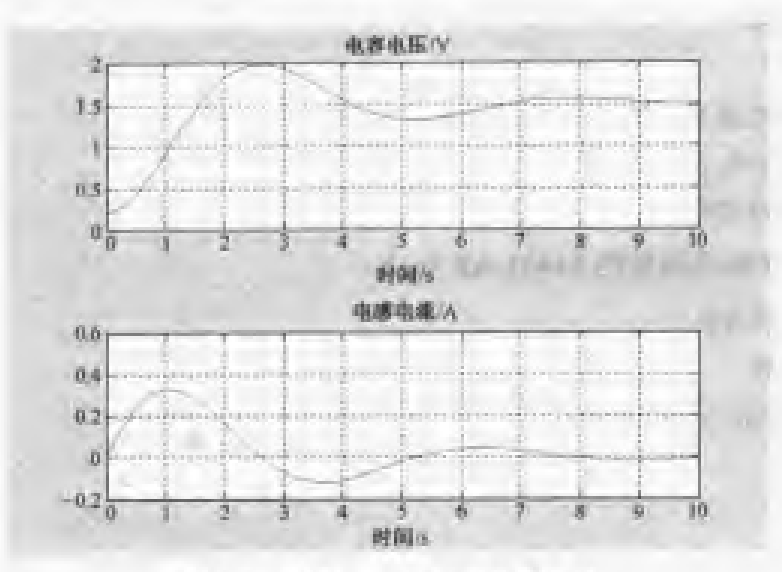


图 4.3　$i(t)$和 $u_c(t)$的时间关系曲线

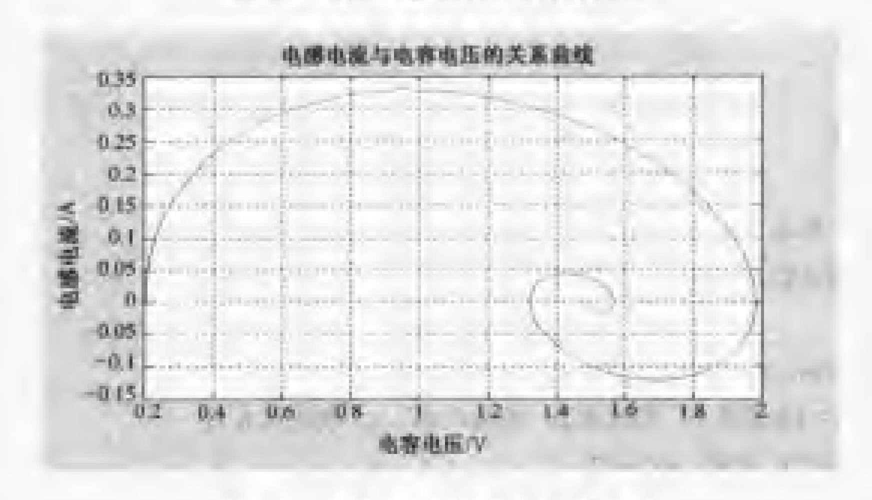


图 4.4　电流与电容电压的关系曲线

4.3　拉氏变换与控制系统模型

动态系统数学模型有多种表达形式，可以是微分方程、差分方程，也可以是传递函数、状态方程。微分方程描述的系统模型，通过求解微分方程，可以得到系统随时间变化的规律，比较直观。但是，当微分方程阶次较高时，微分方程的求解变得十分困难，不易实现，而采用拉氏变换就能把问题的求解从原来的时域变换到复频域，把微分方程变成代数方程，而代数方程的求解通常是比较简单的，求解代数方程后，再通过拉氏反变换得到微分方程的解，两者的关系及运算过程如图 4.5 所示。

时域函数 $f(t)$ 的拉氏变换定义如下：

$$F(s)=\int_{0}^{\infty} f(t)\mathrm{e}^{-st}\mathrm{d}t \tag{4-2}$$

用符号表示为 $F(s)=\mathfrak{L}[f(t)]$，s 称为拉氏算子，它的单位是 1/时间，即频率。由于 s 是复数，因此它还表示复频域变量。拉氏变换的基本定理见表 4.1，常用函数的拉氏变

换见表 4.2。

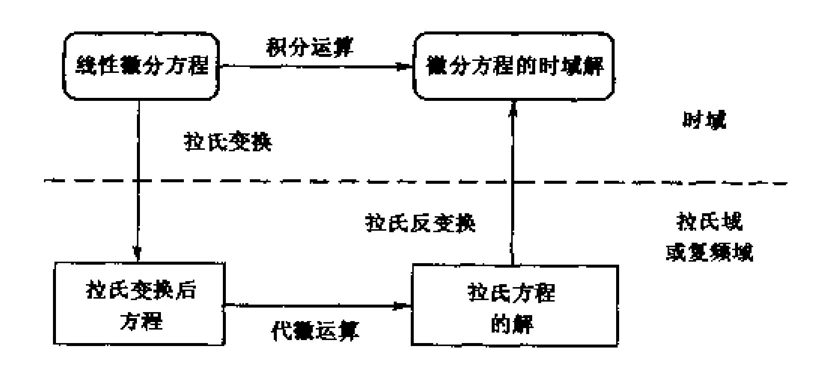


图 4.5　拉氏变换和拉氏反变换

表 4.1　拉氏变换的相关定理

定　理	原函数 $f(t)$	象函数 $F(s)$
线性定理	$af_1(t)+bf_2(t)$， a,b 为实数	$aF_1(s)+bF_2(s)\frac{1}{2}$
复平移定理	$e^{\pm at}f(t)$	$F(s\pm a)$
复实平移定理	$f(t-T)$	$e^{-Ts}F(s)\ (T\geqslant 0)$
标尺定理	$f\left(\frac{t}{a}\right)$	$aF(as)$
微分定理	$\frac{d}{dt}f(t)$	$sF(s)-f(0)$
微分定理	$\frac{d^n}{dt^n}f(t)$	$s^nF(s)-\sum_{r=1}^{n}\frac{d^{r-1}}{dt^{r-1}}f(0)s^{n-r}$
积分定理	$\int_0^t f(t)dt$	$\frac{1}{s}F(s)$
卷积定理	$\int_0^t f_1(\tau)f_2(\tau-t)d\tau$	$F_1(s)F_2(s)$
初值定理	$\lim\limits_{t\to 0}[f(t)]=\lim\limits_{s\to\infty}[sF(s)]=f(0)$	
终值定理	$\lim\limits_{t\to\infty}[f(t)]=\lim\limits_{s\to 0}[sF(s)]=f(\infty)$	

时域函数经过拉氏变换后得到拉氏函数，拉氏反变换定义为：

$$f(t)=\frac{1}{2\pi j}\int_{\sigma-j\omega}^{\sigma+j\omega}F(s)e^{st}ds \tag{4-3}$$

用符号表示为 $f(t)=\mathcal{L}^{-1}[F(s)]$。

直接对 $F(s)$ 积分来计算 $f(t)$ 是复杂的，通常可通过拉氏变换表查找。

在 MATLAB 中，可以采用符号运算工具箱进行拉氏变换和拉氏反变换，采用的函数是 laplace 和 ilaplace，使用前，用 syms 函数设置有关的符号变量。

在 MATLAB 的符号工具箱中，有拉氏变换和拉氏反变换的运算函数，下面进行简单介绍。

符号变量设置函数 syms 的格式为：syms arg1 arg2…，用于设置符号运算中的变量 arg1、arg2 等。也可以用格式：arg1=sym('arg1')；arg2=sym('arg2')；…来进行符号变量设置。

当需要说明变量的数据类型时，可采用以下格式：syms arg1 arg2...datatype。其中 datatype 可以是实型（real）、整型（positive）、非实型（unreal）等。

laplace 变换函数的格式为：L＝laplace(F)。F 是时域函数表达式，约定的自变量是 t，得到的拉氏变换函数是 L(s)。

ilaplace 拉氏反变换函数的格式为：F＝ilaplace(L)，用于把 L 拉氏函数变换为时域函数 F。

表 4.2　常用函数的拉氏变换

$f(t)$	$F(s)$	$f(t)$	$F(s)$
$\delta(t)$ 单位脉冲	1	t	$\frac{1}{s^2}$
$u(t)$ 单位阶跃	$\frac{1}{s}$	t^n	$\frac{n!}{s^{n+1}}$
e^{at}	$\frac{1}{(s-a)}$	$t^n e^{at}$	$\frac{n!}{(s+a)^{n+1}}$
$\cos(\omega t)$	$\frac{s}{(s^2+\omega^2)}$	$\sin(\omega t)$	$\frac{\omega}{(s^2+\omega^2)}$
$e^{-at}\cos(\omega t)$	$\frac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\sin(\omega t)$	$\frac{\omega}{(s+a)^2+\omega^2}$
$\frac{1}{b-a}(e^{-at}-e^{-bt})$	$\frac{1}{(s+a)(s+b)}$	$\frac{1}{b-a}(be^{-bt}-ae^{-at})$	$\frac{s}{(s+a)(s+b)}$

【例 4-3】 求 $\begin{bmatrix} \delta(t-a) & u(t-b) \\ e^{-at}\sin bt & t^2\cos 3t \end{bmatrix}$ 的拉氏变换。

解：MATLAB 程序代码如下。

```
syms t s;
syms a b positive
Dt=sym('Dirac(t-a)');
Ut=sym('Heaviside(t-b)');
Mt=[Dt, Ut;exp(-a*t)*sin(b*t), t^2*exp(-t)];
MS—laplace(Mt, t, s)
```

程序运行后，输出的结果为：

```
MS =
[ exp(–a*s),          exp(–b*s)/s]
[ b/((s+a)^2+b^2),          2/(1+s)^3]
```

4.4　动态过程的传递函数描述

传递函数是在拉氏变换的基础上，以系统本身的参数所描述的线性定常系统输入量和输出量的关系式，它表达了系统内在的固有特性，而与输入量或驱动函数无关。它可

以是有量纲的，也可以是无量纲的，视系统的输入量、输出量而定，它包含着联系输入量与输出量所需要的量纲。它通常不能表明系统的物理特性和物理结构，许多物理性质不同的系统却有着相同的传递函数，正如一些不同的物理现象可以用相同的微分方程描述一样。

4.4.1 传递函数定义与性质

线性定常系统的传递函数定义为：在零初始条件下，输出量（响应函数）的拉普拉斯变换与输入量（驱动函数）的拉普拉斯变化之比。

考虑由下列微分方程描述的线性定常系统：

$$\begin{aligned}&a_0x_o^{(n)}(t)+a_1x_o^{(n-1)}(t)+\cdots+a_{n-1}x_o^{(1)}(t)+a_nx_o(t)\\&=b_0x_i^{(m)}(t)+b_1x_i^{(m-1)}(t)+\cdots+b_{m-1}x_i^{(1)}(t)+b_mx_i(t),\quad n\geqslant m\end{aligned}\tag{4-4}$$

式中，x_o 为系统的输出量，x_i 为系统的输入量，在零初始条件下，输入量与输出量的拉普拉斯变换之比，就是这个系统的传递函数：

$$G(s)=\frac{X_o(s)}{X_i(s)}=\frac{b_0s^m+b_1s^{m-1}+\cdots+b_{m-1}s+b_m}{a_0s^n+a_1s^{n-1}+\cdots+a_{n-1}s+a_n}\tag{4-5}$$

利用传递函数的概念，可以用以 s 为变量的代数方程表示系统的动态特性。如果传递函数分母中 s 的最高次数为 n，则称该系统为 n 阶系统。

在自动控制中，传递函数是个非常重要的概念，它是分析线性定常系统的有力数学工具，它具有以下特点：

- 传递函数比微分方程简单，通过拉氏变换，实数域内复杂的微积分运算转化成代数运算；
- 当系统输入典型信号时，其输出与传递函数有一定对应关系，当输入是单位脉冲函数时，输入的象函数是 1，其输出象函数与传递函数相同；
- 令传递函数中的 $s=j\omega$，则系统可在频域内分析；
- 传递函数的零极点分布决定系统的动态特性。

对线性定常系统，式(4-5)中 s 的系数均为常数，且 a_0 不等于 0，这时系统在 MATLAB 中可以方便地由分子和分母系数构成的两个向量惟一地确定出来，这两个向量分别用 num 和 den 表示。

$$\text{num}=[b_0,b_1,\ldots,b_m],\qquad \text{den}=[a_0,a_1,\ldots,a_n]$$

注意：它们都是按 s 的降幂进行排列的。

在 MATLAB 中，函数 tf 用来建立传递函数模型，即 tf 对象；函数 zpk 用来建立零极点传递函数模型，与建立传递函数模型相关的函数有 tf、tfdata、zpk、zpkdata 和 pzmap，其调用格式如下。

- G = tf(num, den)：num 是分子多项式系数行向量，den 是分母多项式系数行向量。
- [tt, ff] = tfdata(G)：提取 tf 对象模型中的分子分母多项式。

- GG = zpk(G)：将传递函数 tf 对象转换成零极点模型。
- [z, p, k] = zpkdata(G)：提取零极点模型对象的零极点及增益项。
- pzmap(G)：绘制零极点图。

【例 4-4】 某一以微分方程描述系统的传递函数，其微分方程描述如下：

$$y^{(3)}+11y^{(2)}+11y^{(1)}+10y=u^{(2)}+4u^{(1)}+8u$$

试使用 MATLAB 建立其模型。

解：建立模型的 MATLAB 代码如下。

```
%分子多项式系数行向量
num=[1, 4, 8]
%分母多项式系数行向量
den=[1, 11, 11, 10]
%建立传递函数模型
G=tf(num, den)
%显示 tf 对象的特性
get(G)
```

程序运行结果如下：

```
Transfer function:
      s^2 + 4 s + 8
----------------------
s^3 + 11 s^2 + 11 s + 10
tf 对象特性如下：
num: {[0 1 4 8]}
den: {[1 11 11 10]}
Variable: 's'
Ts: 0
ioDelay: 0
InputDelay: 0
OutputDelay: 0
InputName: {''}
OutpntName: {''}
InputGroup: {0x2 cell}
OutputGroup: {0x2 cell}
Notes: {}
UserData: []
```

4.4.2 传递函数零极点表示

零极点模型实际上是传递函数模型的另一种表现形式，其原理是分别对原系统传递函数的分子、分母进行分解因式处理，以获得系统零点和极点的表示形式。

$$G(s)=K\frac{(s-z_1)(s-z_2)\dots(s-z_m)}{(s-p_1)(s-p_2)\dots(s-p_n)}=\frac{K\prod_{i=1}^{m}(s+z_i)}{\prod_{j=1}^{n}(s+p_j)} \tag{4-6}$$

式中，K 为系统增益；$-z_i$ $(i=1,\cdots,m)$ 是分子多项式的根，称为系统的零点：$-p_j$ $(j=1,\cdots,n)$ 是分母多项式的根，称为系统的极点。传递函数的分母多项式就是它的特征多项式，它等于零的方程就是传递函数的特征方程，特征方程的根也就是传递函数的极点。

传递函数的极点决定了所描述系统的自由运动模态，零点影响系统各模态在系统响应中所占的比重，这还会在后面章节详细论述。

在 MATLAB 中零极点增益模型用[z, p, k]矢量组表示，即

$$z = [z_1, z_2, \dots, z_m],\ p=[p_1, p_2, \dots, p_n],\ k=[K]$$

函数 tf2zp( )可以用于求解传递函数的零极点和增益。

【例 4-5】 已知某系统的传递函数 $G(s)=\dfrac{s^2+4s+8}{s^3+11s^2+11s+10}$，求其分子分母多项式并绘制零极点图。

解：MATLAB 程序代码如下。

```
%传递函数分子多项式系数行向量
num=[1, 4, 8]
%传递函数分母多项式系数行向量
den=[1, 11, 11, 10]
%建立传递函数模型
G=tf(num, den)
%提取传递函数的分子和分母多项式
[tt, ff]=tfdata(G, 'v')
%提取传递函数的零极点和增益
[z, p, k]=tf2zp(num, den)
%绘制零极点图
pzmap(G)
%打开绘图网格
grid on
```

程序运行结果如下：

```
tt =
    0    1    4    8
ff =
    1   11   11   10
z =
  -2.0000 + 2.0000i
  -2.0000 - 2.0000i
p =
  -10.0000
  -0.5000 + 0.8660i
  -0.5000 - 0.8660i
k =
    1
```

输出图形如图 4.6 所示。

4.4.3 传递函数的部分分式表示

控制系统常用到并联系统，这时就要对系统函数进行分解，使其表现为一些基本控制单元的和的形式，也就是用部分分式表示，如下式所示：

$$G(s)=K_1\frac{(s-z_1)}{(s-p_1)}+K_2\frac{(s-z_2)}{(s-p_2)}+\dots+K_n\frac{(s-z_n)}{(s-p_n)}=\sum_{i=1}^{n}K_i\frac{(s-z_i)}{(s-p_i)} \tag{4-7}$$

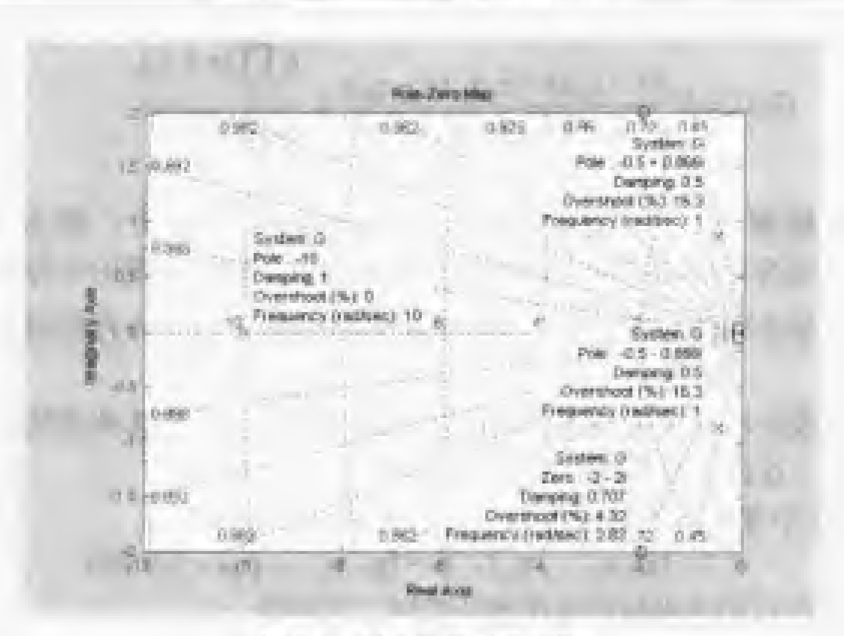


图 4.6　例 4-5 的零极点图

MATLAB 提供了函数[r, p, k]＝residue(b, a)，它的功能是对两个多项式的比进行部分展开，以及把传递函数分解为微分单元的形式。其中向量 b 和 a 是按 s 的降幂排列的多项式系数。部分分式展开后，余数返回到向量 r，极点返回到列向量 p，常数项返回到 k。

[b, a]＝residue(r, p, k)可以将部分分式转化为多项式比 p(s)/q(s)。

【例 4-6】　已知某系统的传递函数 $G(s)=\dfrac{2s^2+9s+1}{s^3+s^2+4s+4}$，求其部分分式表示形式。

解：MATLAB 程序如下。

```
%传递函数分子多项式系数行向量
num=[2, 0, 9, 1];
%传递函数分母多项式系数行向量
den=[1, 1, 4, 4];
%求取系统部分分式表示
[r, p, k]=residue(num, den)
```

程序运行结果如下：

```
r =
   0.0000 - 0.2500i
   0.0000 + 0.2500i
  -2.0000
p =
  -0.0000 + 2.0000i
  -0.0000 - 2.0000i
  -1.0000
```

k =

2

可知结果表达式为：

$$G(s)=2+\frac{-0.25i}{s-2i}+\frac{0.25i}{s+2i}+\frac{-2}{s+1}$$

4.5　动态过程状态空间描述

以传递函数为基础的经典控制理论的数学模型受当时手工计算的局限，着眼于系统的外部联系，重点为单输入单输出的线性定常系统。随着计算机的发展，以状态空间理论为基础的现代控制理论的数学模型则采用状态空间模型，以时域分析为主，着眼于系统的内部状态及其内部联系。

状态是系统动态信息的集合，在表征系统信息的所有变量中，能够全部描述系统运行的最小数目的一组独立变量称为系统的状态变量，其选取不是惟一的。所谓状态方程是由系统状态变量构成的一阶微分方程组。

具有 n 个状态、r 个输入和 m 个输出的线性时不变系统，用矩阵符号表示的状态空间模型如下：

$$\dot{x}=Ax+Bu \text{（状态方程）} \tag{4-8}$$

$$y=Cx+Du \text{（输出方程）} \tag{4-9}$$

式中，状态向量 x 是 n 维的，输入向量 u 是 r 维的，输出向量 y 是 m 维的，状态矩阵 $\boldsymbol{A}$ 是 $n\times n$ 维的，输入矩阵 B 是 $n\times r$ 维的，输出矩阵 C 是 $m\times n$ 维的，前馈矩阵 D 是 $m\times r$ 维的，对于一个时不变系统，$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$、D 都是常数矩阵，如下所示：

$x=[x_1,x_2,\ldots,x_n]^{\mathrm{T}}$，$n$ 维状态向量；

$u=[u_1,u_2,\ldots,u_r]^{\mathrm{T}}$，$r$ 维控制向量；

$y=[y_1,y_2,\ldots,y_m]^{\mathrm{T}}$，$m$ 维输出向量；

$$\boldsymbol{A}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$，$n\times n$ 维系统状态系数矩阵；

$$\boldsymbol{B}=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nr} \end{bmatrix}$$，$n\times r$ 维系统控制系数矩阵；

$$\boldsymbol{C}=\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}$$，$m\times n$ 维输出状态系数矩阵；

$$\boldsymbol{D}=\begin{bmatrix} d_{11} & d_{12} & \dots & d_{1r} \\ d_{21} & d_{22} & \dots & d_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ d_{m1} & d_{m2} & \dots & d_{mr} \end{bmatrix}$$，$m\times r$ 维输出控制系数矩阵。

在 MATLAB 中，系统状态空间用（A, B, C, D）矩阵组表示，当输入（A, B, C, D）矩阵组后，用函数 ss（A, B, C, D）直接可以得到状态空间模型。

【例 4-7】 已知某两输入两输出系统的状态方程和输出方程如下：

$$\dot{x}=\begin{bmatrix} 1 & 6 & 9 & 10 \\ 3 & 12 & 6 & 8 \\ 4 & 7 & 9 & 11 \\ 5 & 12 & 13 & 14 \end{bmatrix}x+\begin{bmatrix} 4 & 6 \\ 2 & 4 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}u,\quad y=\begin{bmatrix} 0 & 0 & 2 & 1 \\ 8 & 0 & 2 & 2 \end{bmatrix}x$$，求其状态空间模型。

解：MATLAB 程序代码如下。

```
%状态矩阵
A=[1 6 9 10; 3 12 6 8; 4 7 9 11; 5 12 13 14];
%输入矩阵
B=[4 6; 2 4; 2 2; 1 0];
%输出矩阵
C=[0 0 2 1; 8 0 2 2];
%前馈矩阵
D=zeros(2, 2);
%生成状态空间模型
G=ss(A, B, C, D)
```

程序运行结果如下：

```
a =
        x1  x2  x3  x4
   x1    1   6   9  10
   x2    3  12   6   8
   x3    4   7   9  11
   x4    5  12  13  14
b =
        u1  u2
   x1    4   6
   x2    2   4
   x3    2   2
   x4    1   0
c =
        x1  x2  x3  x4
   y1    0   0   2   1
   y2    8   0   2   2
```

```
d =
        u1  u2
    y1   0   0
    y2   0   0
```

4.6 系统模型转换及连接

4.6.1 模型转换

线性时不变系统（LTI）的模型包括传递函数（Transfer Function）模型、零极点增益（ZPK）模型和状态空间（State Space）模型，在一些场合下需要用到某种模型，而在另外一些场合下可能需要另外一种模型，这就需要进行模型的转换，它们之间的相互转化关系如图 4.7 所示。

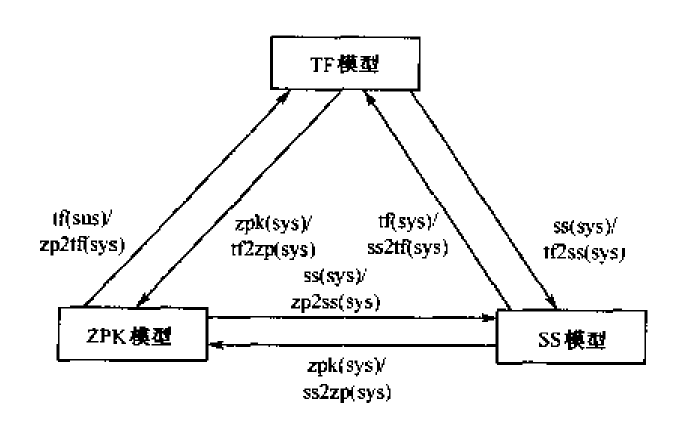


图 4.7 模型转换关系图

MATLAB 提供了丰富的模型转换函数，如表 4.3 所示。

表 4.3 模型转换函数

函数名	功能
residue	传递函数模型与部分分式模型互换
ss2tf	状态空间模型转换为传递函数模型
ss2zp	状态空间模型转换为零极点增益模型
tf2ss	传递函数模型转换为状态空间模型
tf2zp	传递函数模型转换为零极点增益模型
zp2ss	零极点增益模型转换为状态空间模型
zp2tf	零极点增益模型转换为传递函数模型

【例 4-8】 已知某系统的传递函数为 $G(s)=\dfrac{2s^2+9s+1}{s^3+s^2+4s+4}$，试求其零极点模型和状态空间模型。

解：MATLAB 程序代码如下。

```
%传递函数分子多项式系数行向量
num=[2, 9, 1]
%传递函数分母多项式系数行向量
den={1, 1, 4, 4}
%建立传递函数模型
G=tf(num, den)
%将传递函数模型转换成零极点模型
gzpk=zpk(G)
[z, p, k]=zpkdata(G, 'v')
%将传递函数模型转换成状态空间模型
gs=ss(G)
```

程序运行结果如下：

```
Transfer function:
  2 s^2 + 9 s + 1
-------------------
s^3 + s^2 + 4 s + 4
Zero/pole/gain:
2 (s+4.386) (s+0.114)
---------------------
  (s+1) (s^2   + 4)
 z  =
    -4.3860
    -0.1140
p  =
   -0.0000 + 2.0000i
   -0.0000 - 2.0000i
   -1.0000
k  =
       2
%状态空间模型：
a  =
          x1   x2   x3
    x1    -1   -1   -1
    x2     4    0    0
    x3     0    1    0
b  =
          u1
    x1     2
```

```
    x2   0
    x3   0
c =
         x1      x2      x3
    y1      1   1.125   0.125
d =
         u1
    y1   0
```

【例 4-9】 已知某系统的零极点模型$G(s)=\dfrac{6(s+2)}{(s+1)(s+3)(s+5)}$，试求其传递函数模型和状态空间模型。

解：MATLAB 程序代码如下。

```
%系统的零点向量
z=[-2]
%系统的极点向量
p=[-1, -3, -5]
%系统的增益
k=6
%将零极点模型转换成传递函数模型
[num, den]=zp2tf(z, p, k)
%将零极点模型转换成状态空间模型
[A, B, C, D]=zp2ss(z, p, k)
%建立零极点模型
g_zpk=zpk(z, p, k)
%建立传递函数模型
g_tf=tf(num, den)
%建立状态空间模型
g_ss=ss(A, B, C, D)
```

程序运行结果如下：

```
Zero/pole/gain:
      6 (s+2)
-----------------
(s+1) (s+3) (s+5)
Transfer function:
        6 s + 12
----------------------
s^3 + 9 s^2 + 23 s + 15
a =
```

```
              x1        x2        x3
    x1        -1         0         0
    x2         1        -8    -3.873
    x3         0     3.873         0
b =
          u1
    x1    1
    x2    1
    x3    0
c =
              x1        x2        x3
    y1         0         0     1.549
d =
          u1
    y1    0
Continuous-time model
```

4.6.2 模型连接

实际应用中，整个自动控制系统是由多个单一的模型组合而成的。模型之间有不同的连接方式，基本的连接方式有并联、串联、反馈和闭环连接，下面分别进行论述。

4.6.2.1 串联连接

单输入单输出（SISO）系统 $G_1(s)$ 和 $G_2(s)$ 串联连接的结构框图如图 4.8 所示，$G_1(s)$ 和 $G_2(s)$ 串联连接合成的系统的传递函数为 $G(s)=G_1(s)\cdot G_2(s)$。

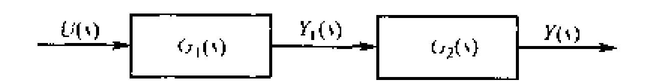


图 4.8 串联连接结构示意图

MATLAB 提供了进行模型串联的函数 series，其格式如下：

[a, b, c, d]=series(a1, b1, c1, d1, a2, b2, c2, d2)，表示串联连接两个状态空间系统。

[a, b, c, d]=series(a1, b1, c1, d1, a2, b2, c2, d2, out1, in2)，表示 out1 和 in2 分别指定系统 1 的部分输出和系统 2 的部分输入进行连接。

[num, den]=series(num1, den1, num2, den2)，表示将串联连接的传递函数进行相乘。

【例 4-10】 已知两系统的传递函数 $G_1(s)=\dfrac{6(s+2)}{(s+1)(s+3)(s+5)}$，$G_2(s)=\dfrac{(s+2.5)}{(s+1)(s+4)}$，试求两系统串联的传递函数。

解：MATLAB 程序代码如下。

```
%传递函数 1 的分子多项式系数行向量
```

```
num1=[6, 12]
%传递函数 1 的分母多项式系数行向量
den1=[1, 9, 23, 15]
%传递函数 2 的分子多项式系数行向量
num2=[1, 2.5]
%传递函数 2 的分母多项式系数行向量
den2=[1, 5, 4]
%串联连接
[num, den]=series(num1, den1, num2, den2)
g_tf=tf(num, den)
```

程序运行结果如下：

```
num =
     0     0     0     6     27     30
den =
     1    14    72   166    167     60
Transfer function:
                  6 s^2 + 27 s + 30
-------------------------------------------
s^5 + 14 s^4 + 72 s^3 + 166 s^2 + 167 s + 60
```

由计算结果可知，$G_1(s)$ 和 $G_2(s)$ 两系统串联组成的传递函数为：

$$\frac{6s^2+27s+30}{s^5+14s^4+72s^3+166s^2+167s+60}$$

4.6.2.2　并联连接

单输入单输出（SISO）系统 $G_1(s)$ 和 $G_2(s)$ 并联连接的结构框图如图 4.9 所示，$G_1(s)$ 和 $G_2(s)$ 并联连接合成的系统的传递函数为 $G(s)=G_1(s)+G_2(s)$ 。

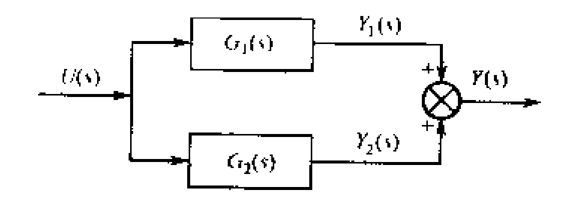


图 4.9　并联连接结构示意图

MATLAB 提供了进行模型并联的函数 parallel，它的格式如下：

[a, b, c, d] = parallel(a1, b1, c1, d1, a2, b2, c2, d2)，表示并联连接两个状态空间系统。

[a, b, c, d] = parallel(a1, b1, c1, d1, a2, b2, c2, d2, inp1, inp2, out1, out2)，其中 inp1 和 inp2 分别指定两系统中要连接在一起的输入端编号，从 u1, u2, …, un 依次编号为 1, 2, …, n;

out1 和 out2 分别指定要相加的输出端编号，编号方式与输入类似。inp1 和 inp2 既可以是标量也可以是向量。out1 和 out2 用法与之相同。例如：inp1＝1，inp2＝3 表示系统 1 的第一个输入端与系统 2 的第三个输入端相连接。若 inp1＝[1 3], inp2＝[2 1]则表示系统 1 的第一个输入与系统 2 的第二个输入连接，以及系统 1 的第三个输入与系统 2 的第一个输入连接。

[num, den]＝parallel(num1, den1, num2, den2)，表示将并联连接的传递函数进行相加。

【例 4-11】 已知两系统的传递函数 $G_1(s)=\dfrac{6(s+2)}{(s+1)(s+3)(s+5)}$，$G_2(s)=\dfrac{(s+2.5)}{(s+1)(s+4)}$，试求两系统并联的传递函数。

解：MATLAB 程序代码如下。

```
%传递函数 1 的分子多项式系数行向量
num1=[6, 12]
%传递函数 1 的分母多项式系数行向量
den1=[1, 9, 23, 15]
%传递函数 2 的分子多项式系数行向量
num2=[1, 2.5]
%传递函数 2 的分母多项式系数行向量
den2=[1, 5, 4]
%并联连接
[num, den]=parallel(num1, den1, num2, den2)
g_tf=tf(num, den)
```

程序运行结果如下：

```
num =
          0    1.0000    17.5000    87.5000   156.5000    85.5000
den =
      1    14    72    166    167    60
Transfer function:
 s^4 + 17.5 s^3 + 87.5 s^2 + 156.5 s + 85.5
--------------------------------------------
s^5 + 14 s^4 + 72 s^3 + 166 s^2 + 167 s + 60
```

由计算结果可知，$G_1(s)$ 和 $G_2(s)$ 两系统并联组成的传递函数为：

$$\frac{s^4+17.5s^3+87.5s^2+156.5s+85.5}{s^5+14s^4+72s^3+166s^2+167s+60}$$

4.6.2.3 反馈连接

反馈系统在自动控制中是应用最为广泛的系统。最常用的反馈连接是将系统 $G(s)$ 的全部输出信号反馈作为另一个系统 $H(s)$ 的输入，根据 $H(s)$ 输出与 $G(s)$ 输入信号之间是相加还是相减，系统分为正反馈或负反馈，一般情况下是如图 4.10 所示的反馈系统，其中，$G(s)$ 称为前向传递函数，$H(s)$ 称为反馈传递函数。

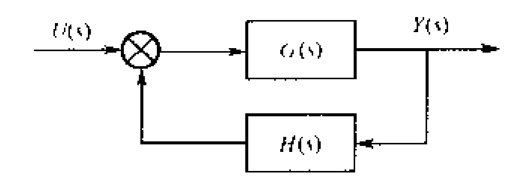


图 4.10　反馈连接结构示意图

当系统正反馈连接时，合成的系统传递函数 $GH(s)$ 为：

$$GH(s)=\frac{G(s)H(s)}{1-G(s)H(s)} \tag{4-10}$$

当系统负反馈连接时，合成的系统传递函数 $GH(s)$ 为：

$$GH(s)=\frac{G(s)H(s)}{1+G(s)H(s)} \tag{4-11}$$

MATLAB 提供了进行模型反馈连接的函数 feedback，其格式如下：

[a, b, c, d]＝feedback(a1, b1, c1, d1, a2, b2, c2, d2)，表示将两个系统按反馈方式连接，一般而言，系统 1 为对象，系统 2 为反馈控制器。

[a, b, c, d]＝feedback(a1, b1, c1, d1, a2, b2, c2, d2, sign)，表示系统 1 的所有输出连接到系统 2 的输入，系统 2 的所有输出连接到系统 1 的输入，sign 用来指示系统 2 输出到系统 1 输入的连接符号，sign，默认为负值，即 sign＝ −1。总系统的输入/输出数等同于系统 1。

[a, b, c, d]＝feedback(a1, b1, c1, d1, a2, b2, c2, d2, inp1, out1)，表示部分反馈连接，将系统 1 的指定输出 out1 连接到系统 2 的输入，系统 2 的输出连接到系统 1 的指定输入 inp1，以此构成闭环系统。

[num, den]＝feedback(num1, deu1, nnm2, den2, sign)，表示可以得到类似的连接，只是子系统和闭环系统均以传递函数的形式表示。sign 的含义与前述相同。

【例 4-12】　已知系统的前向传递函数 $G_1(s)=\dfrac{s-1}{s^2-5s-2}$，反馈传递函数 $H(s)=\dfrac{s+1}{s^2+3s+2}$，试求它们组成的负反馈传递函数。

解：MATLAB 程序代码如下。

```
%前向传递函数的分子多项式系数行向量
num1=[1, -1]
%前向传递函数的分母多项式系数行向量
den1=[1, -5, -2]
%反馈传递函数的分子多项式系数行向量
num2=[1, 1]
%反馈传递函数的分母多项式系数行向量
den2=[1, 3, 2]
%反馈连接
[num, den]=feedback(num1, den1, num2, dea2)
%建立传递函数模型
```

```
tf(num, den)
```

程序运行结果如下：

```
num =
     0     1     2    -1    -2
den =
     1    -2   -14   -16    -5
Transfer function:
      s^3 + 2 s^2 - s - 2
--------------------------------
s^4 - 2 s^3 - 14 s^2 - 16 s - 5
```

由计算结果可知，$G_1(s)$ 和 $H(s)$ 组成的负反馈传递函数为 $\dfrac{s^3+2s^2-s-2}{s^4-2s^3-14s^2-16s-5}$。

4.6.2.4 闭环连接

这里的闭环连接指的是通常意义下的单位反馈连接，这种连接方式在实际中大量存在，它的连接方式属于反馈连接，是反馈传递函数 $H(s)=1$ 的一个特例，连接形式如图 4.11 所示。

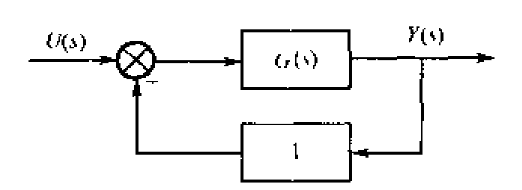


图 4.11　反馈连接结构示意图

MATLAB 提供了进行闭环连接的函数 cloop，其格式如下：

[ac, bc, cc, dc]=cloop(a, b, c, d, sign)，表示通过将所有的输出反馈到输入，从而产生闭环系统的状态空间模型。当 sign=1 时采用正反馈；当 sign=–1 时采用负反馈；sign 缺省时默认为负反馈。

[ac, bc, cc, dc]=cloop(a, b, c, d, outputs, inputs)，表示将指定的输出 outputs 反馈到指定的输入 inputs，以此构成闭环系统的状态空间模型。一般为正反馈，形成负反馈时应在 inputs 中采用负值。

[uumc, denc]=cloop(num, den, sign)，表示由传递函数表示的开环系统构成闭环系统，sigu 意义与上述相同。

【例 4-13】 已知系统的前向传递函数 $G_1(s)=\dfrac{s-1}{s^2-5s-2}$，试求闭环连接的传递函数。

解：MATLAB 程序代码如下。

```
%前向传递函数的分子多项式系数行向量
num1=[1, -1]
%前向传递函数的分母多项式系数行向量
den1=[1, -5, -2]
```

```
%闭环连接
[num, den]=cloop(num1, den1)
%建立传递函数模型
tf(num, den)
```

程序运行结果如下：

```
num =
     0     1    -1
den =
     1    -4    -3
Transfer function:
   s - 1
-------------
s^2 - 4 s - 3
```

由计算结果可知，闭环连接的传递函数为 $\dfrac{s-1}{s^2-4s-3}$。

4.7　非线性数学模型的线性化

系统如果不能应用叠加原理，则系统是非线性的。在建立控制系统的微分方程时，常常遇到非线性方程。由于解非线性微分方程比较困难，因而提出了非线性特性的线性化问题。如果我们能够做某种近似，或者缩小一些研究问题的范围，那么大部分非线性特性都可以近似地作为线性特性来处理，这会给控制系统研究工作带来诸多方便。虽然这种方法是近似的，但在一定范围内能够反映系统的特性，在工程实践中有很大的实际意义。

在控制工程中，如果系统的运行是围绕平衡点进行的，并且系统中的信号是围绕平衡点变化的小信号，那么就可以用线性系统去近似非线性系统。这种线性系统在有限的工作范围内等效于原来的非线性系统。在控制工程中，这种线性化模型（线性定常模型）是很重要的。

线性化过程用数学方法来处理就是将一个非线性函数 $y=f(x)$，在其工作点（x_0，y_0）处展开成泰勒级数，然后忽略其二次以上的高阶项得到线性化方程，并以次代替原来的非线性函数。因为忽略了泰勒级数展开中的高阶项，所以这些被忽略的项必须很小，即变量只能对工作状态有微小的偏离。

对于具有一个自变量的非线性函数，设其输入量为 $x(t)$，输出量为 $y(t)$，系统正常工作点为 $y_0=f(x_0)$，那么在 $y_0=f(x_0)$ 附近展开成泰勒级数为：

$$y=f(x_0)+\left(\frac{\mathrm{d}f(x)}{\mathrm{d}x}\right)_{x=x_0}(x-x_0)+\frac{1}{2!}\left(\frac{\mathrm{d}^2f(x)}{\mathrm{d}x^2}\right)_{x=x_0}(x-x_0)^2+\ldots \tag{4-12}$$

式中的导数 $\dfrac{\mathrm{d}f(x)}{\mathrm{d}x}$，$\dfrac{\mathrm{d}^2f(x)}{\mathrm{d}x^2}$ …均是在 $x=x_0$ 点上计算得到的。如果变量的变化 $x-x_0$ 很小，

则可以忽略二次以上的项，可写成：

$$y = f(x_0) + (\frac{df(x)}{dx})_{x=x_0}(x - x_0) \tag{4-13}$$

或

$$y = y_0 + K(x - x_0) \tag{4-14}$$

其中，$y_0 = f(x_0)$，$K = (\frac{df(x)}{dx})_{x=x_0}$

式（4-14）表明 $y - y_0$ 与 $x - x_0$ 成正比，方程（4-14）就是方程 $y_0 = f(x_0)$ 在工作点 (x_0, y_0) 附近的线性化模型。

对于多输入量函数的线性化，下面以两个输入变量的函数 $y = f(x_1, x_2)$ 在工作点 $x_1 = x_{10}$，$x_2 = x_{20}$ 处的线性化为例进行介绍。

方程 $y = f(x_1, x_2)$ 在工作点附近展开成泰勒级数如下：

$$\begin{aligned} y = & f(x_{10}, x_{20}) + [(\frac{\partial f}{\partial x_1})(x_1 - x_{10}) + (\frac{\partial f}{\partial x_2})(x_2 - x_{20})] \\ & + \frac{1}{2!}[(\frac{\partial^2 f}{\partial x_1^2})(x_1 - x_{10})^2 + 2(\frac{\partial^2 f}{\partial x_1 \partial x_2})(x_1 - x_{10})(x_2 - x_{20}) + (\frac{\partial^2 f}{\partial x_2^2})(x_2 - x_{20})^2] + \dots \end{aligned} \tag{4-15}$$

式中的偏导数都在 $x_1 = x_{10}$，$x_2 = x_{20}$ 上计算得到，在工作点附近，高阶项可以忽略不计，于是式（4-15）可以写成如下形式：

$$y = f(x_{10}, x_{20}) + (\frac{df}{dx_1})_{x_1=x_{10}}(x_1 - x_{10}) + (\frac{df}{dx_2})_{x_2=x_{20}}(x_2 - x_{20}) \tag{4-16}$$

或者

$$y = y_0 + K_1(x_1 - x_{10}) + K_2(x_2 - x_{20}) \tag{4-17}$$

其中，$K_1 = (\frac{df}{dx_1})_{x_1=x_{10}, x_2=x_{20}}$，$K_2 = (\frac{df}{dx_2})_{x_1=x_{10}, x_2=x_{20}}$

上述线性化方法只有在工作状态附近才是正确的。当工作状态的变化范围很大时，线性化方程就不合适了，这时必须使用非线性方程。应当特别注意，在分析和设计中采用的具体数学模型只是在一定的工作条件下才能精确地表示实际系统的动态特性，在其他工作条件下它可能是不精确的。

第 5 章　时域分析法

5.1　引　　言

时域分析法是以拉普拉斯变换为工具，从传递函数出发，直接在时间域上研究自动控制系统性能的一种方法。这种方法的优点是对系统分析的结果直接而全面，缺点是分析过程的计算量较大，尤其是对高阶系统。一般情况下，系统分析结果所提供的信息对下一步如何改造、综合系统是不够的。计算机仿真技术的发展，尤其是 MATLAB/Simulink 的广泛应用，正好弥补了这一不足。时域分析法是其他分析法的基础，如根轨迹法和频率法；另外，一般来说用根轨迹法和频率法综合的系统最终也需要用时域分析法进行验证。

通过本章，使读者熟悉和掌握时域分析法，并能使用 MATLAB/Simulink 对控制系统进行时域分析。

5.2　时域响应分析

时域响应指的是系统在外部输入（设定值输入或扰动输入）作用下的输出过程。

5.2.1　典型输入

为了研究自动控制系统的暂态特性和稳态特性，需要知道输入量的变化规律，但通常是不能准确知道输入量的变化的。不同的输入形式其响应是不同的，为了便于比较，常用一些规定的输入形式作为系统输入来检查系统的性能，这些输入就称为典型输入。自动控制系统通常使用的典型输入信号有脉冲输入、阶跃输入、斜坡输入、加速度输入和正弦输入。利用这些典型输入信号易于对系统进行试验和数学分析。

1. 单位脉冲输入

$$r(t)=\delta(t)=\begin{cases}\infty, & t=0\\ 0, & t\neq 0\end{cases} \tag{5-1}$$

其中，

$$\int_{-\infty}^{+\infty}\delta(t)\mathrm{d}t=1 \tag{5-2}$$

其拉氏变换是：

$$R(s)=L\left[\delta(t)\right]=1 \tag{5-3}$$

单位脉冲函数的幅值为无穷大，持续时间为零，纯属数学上的假设，但在系统分析中是很有用的。实际中，常用系统受到单位脉冲输入作用后的输出来衡量系统的暂态响应特性。

单位脉冲响应的拉氏变换就是系统的传递函数。如果在系统输入端加一单位脉冲函数，由输出响应即可求得系统的传递函数。

在分析闭环系统时，单位脉冲输入具有重要意义。根据系统的脉冲响应可以求出系统的传递函数，并且可以求出任意输入信号下的系统响应。

2. 单位阶跃输入

$$r(t)=1(t)=\begin{cases}1, & t>0\\ 0, & t\leqslant 0\end{cases} \tag{5-4}$$

其拉氏变换是：

$$R(s)=L\left[1(t)\right]=\frac{1}{s} \tag{5-5}$$

在 $t=0$ 处的阶跃信号相当于一个不变的信号突然加到系统上。对于恒值系统，相当于给定值突然发生变化；对于随动系统，相当于加一个突变的给定位置信号。

3. 单位斜坡输入

$$r(t)=\begin{cases}t, & t>0\\ 0, & t\leqslant 0\end{cases} \tag{5-6}$$

其拉氏变换是：

$$R(s)=L\left[r(t)\right]=\frac{1}{s^2} \tag{5-7}$$

单位斜坡输入对于随动系统，相当于加一个恒速变化的位置信号。

4. 单位加速度输入

$$r(t)=\begin{cases}\frac{1}{2}t^2, & t>0\\ 0, & t\leqslant 0\end{cases} \tag{5-8}$$

其拉氏变换是：

$$R(s)=L\left[r(t)\right]=\frac{2}{s^3} \tag{5-9}$$

单位加速度输入对于随动系统，相当于加一个恒加速度变化的位置信号。

5. 单位正弦输入

$$r(t)=\begin{cases}\sin(t), & t>0\\ 0, & t\leqslant 0\end{cases} \tag{5-10}$$

利用单位正弦输入可以求得系统对不同频率的稳态响应，由之可以间接判断系统性能。

典型输入的选用应视不同系统要求而定。例如，对于恒值控制系统通常用阶跃输入；对于随动系统通常用斜坡输入和加速度输入；对于扰动与响应系统通常用阶跃输入和脉冲输入。

5.2.2　线性系线时域响应一般求法

设已知系统的传递函数为：

$$G(s)=\frac{C(s)}{R(s)}=\frac{k_0(s+z_1)(s+z_2)\ldots(s+z_m)}{(s+p_1)(s+p_2)\ldots(s+p_n)} \tag{5-11}$$

且输入为：

$$R(s)=\frac{k_r(s+z_{r1})(s+z_{r2})\ldots(s+z_{rl})}{(s+p_{r1})(s+p_{r2})\ldots(s+p_{rq})} \tag{5-12}$$

式中，$-z_{r1}$，$-z_{r2}$，…，$-z_{rl}$ 及 $-p_{r1}$，$-p_{r2}$，…，$-p_{rq}$ 分别是输入函数 $r(t)$ 拉氏变换式 $R(s)$ 的零点和极点，通常简称为输入零点和极点。

那么，

$$C(s)=G(s)\cdot R(s)=\frac{k_0k_r(s+z_1)(s+z_2)\ldots(s+z_m)(s+z_{r1})(s+z_{r2})\ldots(s+z_{rl})}{(s+p_1)(s+p_2)\ldots(s+p_n)(s+p_{r1})(s+p_{r2})\ldots(s+p_{rq})} \tag{5-13}$$

5.2.2.1　$G(s)$无重极点的情况

若式（5-13）中无重极点，则 $C(s)$ 可分解为：

$$C(s)=\sum_{i=1}^{q}\frac{A_{ri}}{s+p_{ri}}+\sum_{j=1}^{n}\frac{A_j}{s+p_j} \tag{5-14}$$

那么，

$$c(t)=\sum_{i=1}^{q}A_{ri}\cdot e^{-p_{ri}\cdot t}+\sum_{j=1}^{n}A_j\cdot e^{-p_j\cdot t} \tag{5-15}$$

其中，$A_{ri}=C(s)(s+p_{ri})|_{s=-p_{ri}}$，$A_j=C(s)(s+p_j)|_{s=-p_j}$。

通常把 $c(t)$ 表达式中线性独立的分量 $e^{-p_{ri}\cdot t}$ 和 $e^{-p_j\cdot t}$ 称为模态，它取决于传递函数的极点和输入极点。模态的特征是发散还是收敛，是单调变化还是振荡，决定了响应的基本特征。A_{ri} 和 A_j 称为 $C(s)$ 的留数，它由 $C(s)$ 的极点和零点共同决定。留数的大小和模态共同决定了响应。由此可知，系统时域响应的特征是由闭环极点决定的。零点可以影响 $c(t)$ 的具体变化形状。另一方面，通常把式（5-14）中的第一项称为响应的稳态分量，它和输入极点有关；第二项称为响应的动态分量，它和系统传递函数的极点有关。当然，系统本身的动态性能是由其动态分量决定的。

5.2.2.2　$G(s)$有重极点的情况

$C(s)$有重极点时，方法会有所不同，下面进行简要说明。

设

$$G(s)=\frac{M(s)}{(s+p_1)^m(s+p_{m+1})\ldots(s+p_n)} \tag{5-16}$$

其中，$-p_1$ 是 m 重极点，那么，

$$C(s)=\frac{A_1}{s+p_1}+\cdots+\frac{A_m}{(s+p_1)^m}+\frac{A_{m+1}}{s+p_{m+1}}+\cdots+\frac{A_n}{s+p_n} \tag{5-17}$$

其中，

$$A_i=\begin{cases}\dfrac{1}{(i-1)!}\cdot\dfrac{\mathrm{d}^{i-1}}{\mathrm{d}s^{i-1}}C(s)(s+p_1)^m|_{s=-p_1} & i=1,2,\cdots,m\\ C(s)(s+p_i)|_{s=-p_i} & i=m+1,\cdots n\end{cases} \tag{5-18}$$

而

$$\begin{aligned}c(t)&=A_1\cdot\mathrm{e}^{-p_1\cdot t}+A_2\cdot t\cdot\mathrm{e}^{-p_1\cdot t}+\cdots\\&+\frac{A_m}{(m-1)!}\cdot t^{m-1}\cdot\mathrm{e}^{\ p_1\cdot t}+A_{m+1}\cdot\mathrm{e}^{-p_{m+1}\cdot t}+\cdots+A_n\cdot\mathrm{e}^{-p_n\cdot t}\end{aligned} \tag{5-19}$$

5.2.3 时域响应性能指标

5.2.3.1 阶跃响应性能指标

当已知时域响应 $c(t)$ 时，按 $c(t)$ 的形状就大致可判断出其动力学性能的优劣。一般来说，对系统输出响应的要求可以用两个基本要求和三个衡量标准来概括。

两个基本要求是：对设定值输入的跟随、对扰动输入的抑制。

三个衡量标准是：跟随和抑制过程的稳定性、快速性和准确性。

下面对设定值为阶跃信号的响应加以说明，通常将之称为阶跃响应。阶跃响应的一般情况如图 5.1 所示。

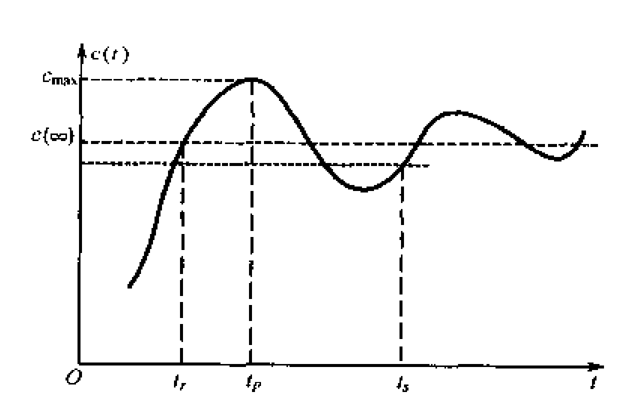


图 5.1　阶跃响应瞬态指标图

图 5.1 中，t_r 称为上升时间，它表示 $c(t)$ 第一次达到稳态值 $c(\infty)$ 的时间。对于单调无超调过程显然该定义不适用，一般可定义为 $c(t)$ 从 $0.1c(\infty)$ 上升到 $0.9c(\infty)$ 所需的时间。

（1）t_p 称为峰值时间，它表示 $c(t)$ 达到最大值 $c_{\max}$ 的时间。

（2）$\sigma_p\%$ 称为超调量，且

$$\sigma_p\%=\frac{c_{\max}-c(\infty)}{c(\infty)}\times100\% \tag{5-20}$$

（3）t_s 称为调节时间，且

$$\left|\frac{c(t)-c(\infty)}{c(\infty)}\right|_{t\geqslant t_s}\leqslant\Delta \tag{5-21}$$

其中，Δ 称为允许误差，一般取 $\Delta=0.05$ 或 $\Delta=0.02$。

上述指数显然从不同的侧面反映了系统的性能，其中 t_r 反映了快速性，$\sigma_p\%$ 反映了稳定性，而 t_s 是稳定性和快速性的一种综合。

5.2.3.2　误差积分指标

上述指标从不同侧面衡量了系统的性能，但在系统分析和最优设计时还需用到一种误差积分指标。

定义误差函数 $e(t)=c(\infty)-c(t)$，如图 5.2 中的阴影部分所示。显然，阴影部分的面积越小则跟随性能越好。

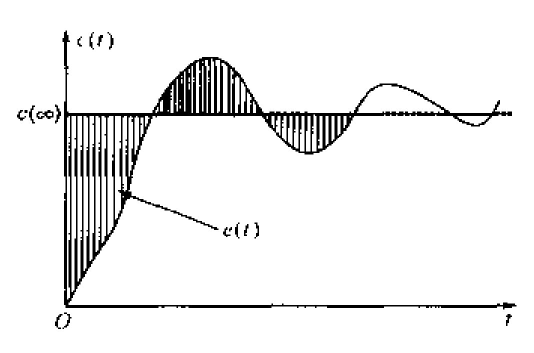


图 5.2　阶跃响应误差指标图

有如下几种常用的误差积分指标。

（1）误差积分指标 IE

$$\mathrm{IE}=\int_0^{\infty}e(t)\mathrm{d}t \tag{5-22}$$

显然，该指标对于振荡型过程不适用。

（2）误差绝对值积分指标 IAE

$$\mathrm{IAE}=\int_0^{\infty}|e(t)|\mathrm{d}t \tag{5-23}$$

（3）误差平方积分指标 ISE

$$\mathrm{ISE}=\int_0^{\infty}e^2(t)\mathrm{d}t \tag{5-24}$$

后两种指标均用于振荡型响应分析，其中 ISE 指标比较适合在理论分析中使用。一般来说，在同等条件下，用 ISE 指标设计出来的系统与用 IAE 指标设计出来的系统相比较，其 t_r 较小而 $\sigma_p\%$ 较大。

（4）误差绝对值乘时间积分指标 ITAE

$$\mathrm{ITAE}=\int_0^{\infty}|e(t)|t\mathrm{d}t \tag{5-25}$$

由图 5.2 可以看出，IAE 和 ISE 指标的大小主要取决于第一块面积，但在一般情况下，

系统的相对稳定性σ_p%和综合快速性t_s由第二块面积的大小来反映，因此应加上第二块在指数函数中的份额，处理方法是乘以时间函数t作为权系数。该指标目前被广泛应用于最优化分析和设计中。

5.2.4 MATLAB/Simulink 在时域分析中的应用

时域分析，尤其是高阶系统的时域分析，其困难主要在系统极点、留数的获取上，以及在已知响应表达式的基础上，如何绘制响应波形和求取性能指标等一系列问题，这些均涉及大量的数值计算，MATLAB/Simulink 的仿真平台为此提供了强有力的工具。

5.2.4.1 时域分析中 MATLAB 函数的应用

一个动态系统的性能常用典型输入作用下的响应来描述。响应是指零初始值条件下某种典型的输入函数作用下对象的响应，控制系统常用的输入函数为单位阶跃函数和脉冲激励函数（即冲激函数）。

在 MATLAB 的控制系统工具箱中提供了求这两种输入下系统响应的函数。常用的函数有单位阶跃响应函数 step( )；冲激响应函数 impulse( )和时域分析函数。

1. step( )函数的用法

y＝step(num, den, t)：其中 num 和 den 分别为系统传递函数描述中的分子和分母多项式系数，t 为选定的仿真时间向量，一般可由 t＝0:step:end 等步长地产生。该函数返回值 y 为系统在仿真中所得输出组成的矩阵。

[y, x, t]＝step(num, den)：时间向量 t 由系统模型特性自动生成, 状态变量 x 返回为空矩阵。

[y, x, t]＝step(A, B, C, D, iu)：其中 A, B, C, D 为系统的状态空间描述矩阵，iu 用来指明输入变量的序号，x 为系统返回的状态轨迹。

如果对具体的响应值不感兴趣，而只想绘制系统的阶跃响应曲线，则可采用以下格式进行函数调用：

```
step(num, den)
step(num, den, t)
step(A, B, C, D, iu, t)
step(A, B, C, D, iu)
```

线性系统的稳态值可以通过函数 dcgain( )来求得，其调用格式为 dc＝dcgain(num, den) 或 dc＝dcgain(a, b, c, d)。

2. impulse( )函数的用法

求取脉冲激励响应的调用方法与 step( )函数基本一致。

```
y＝impnlse(num, den, t)
[y, x, t]＝impulse(num, den)
```

```
[y, x, t]=impulse(A, B, C, D, iu, t)
impulse(num, den)
impulse(num, den, t)
impulse(A, B, C, D, iu)
impulse(A, B, C, D, iu, t)
```

3. 常用时域分析函数

时间响应分析的是系统对输入和扰动在时域内的瞬态行为。系统特征，如上升时间、调节时间、超调量和稳态误差，均能从时间响应上反映出来。

MATLAB 除提供前面介绍的对系统阶跃响应、冲激响应等进行仿真的函数外，还提供大量对控制系统进行时域分析的函数。

covar：连续系统对白噪声的方差响应。

initial：连续系统的零输入响应。

lsim：连续系统对任意输入的响应。

对于离数系统，只需在连续系统对应函数前加"d"即可，如 dstep，dimpulse 等，其调用格式与 step、impulse 类似。

5.2.4.2 MATLAB 在时域分析中应用举例

【例 5-1】 已知系统的闭环传递函数如下：

$$G(s)=\frac{1}{s^2+0.4s+1}$$

试求其单位阶跃响应曲线。

解：MATLAB 程序代码如下。

```
num=[1]
den=[1, 0.4, 1]
t=[0:0.1:10]
[y, x, t]=step(num, den, t)
plot(t, y)
grid
xlabel('Time [sec] t')
ylabel('y')
```

响应曲线如图 5.3 所示。

【例 5-2】 已知系统的闭环传递函数如下：

$$\frac{C(s)}{R(s)}=G(s)=\frac{1}{s^2+0.3s+1}$$

试求其单位斜坡输入响应曲线。

解：MATLAB 程序代码如下。

```
num=[1]
den=[1, 0.3, 1]
t=[0:0.1:10]
u=t;
[y, x]=lsim(num, den, u, t)
plot(t, y)
grid
xlabel('Time [sec] t')
ylabel('y')
```

响应曲线如图 5.4 所示。

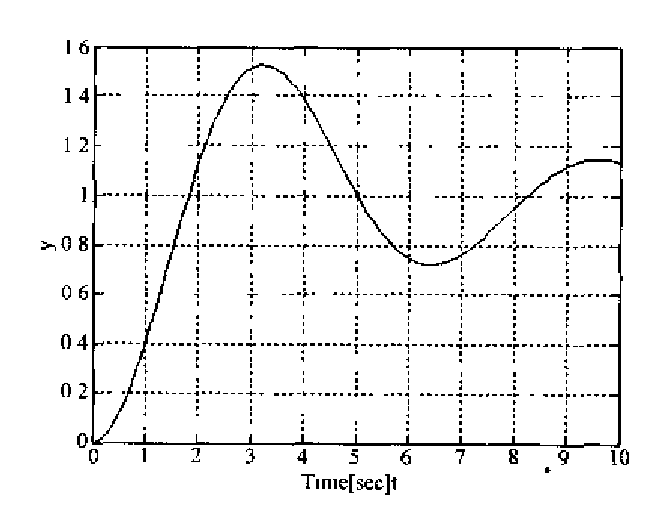


图 5.3　例 5-1 阶跃响应曲线

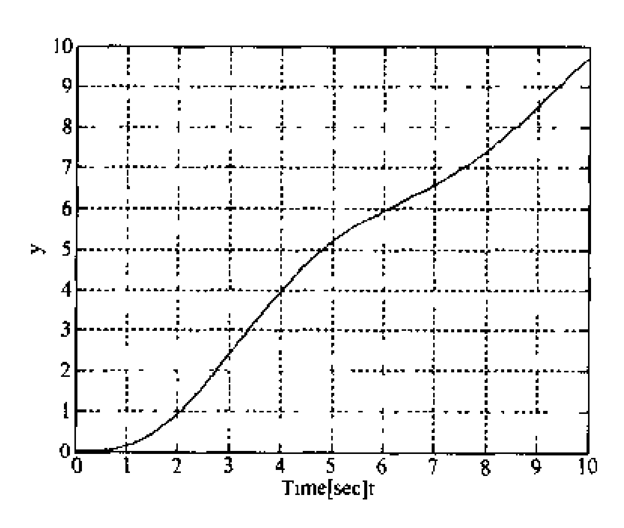


图 5.4　例 5-2 斜坡输入响应曲线

【例 5-3】　反馈系统如图 5.5 所示，其中 $G(s)=\dfrac{s+2}{s^2+10s+1}$，系统输入信号为如图 5.6 所示的三角波，试求取系统输出响应，并将输入输出信号对比显示。

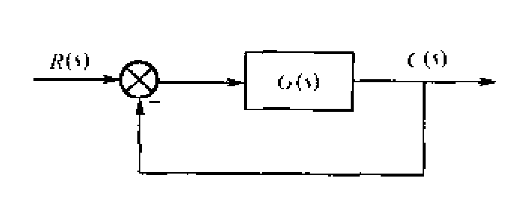


图 5.5　例 5-3 系统框图

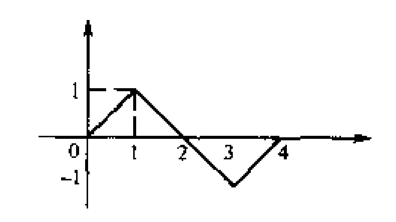


图 5.6　例 5-3 输入曲线

解：MATLAB 程序代码如下。

```
numg=[1, 2]
deng=[1, 10, 1]
[num, den]=cloop(numg, deng, -1)
```

```
v1=[0:0.1:1]
v2=[0.9:-0.1:-1]
v3=[-0.9:0.1:0]
t=[0:0.1:4]
u=[v1, v2, v3]
[y, x]=lsim(num, den, u, t)
plot(t, y, t, u)
xlabel('Time [sec]')
ylabel('theta [rad]')
grid
```

响应曲线如图 5.7 所示。

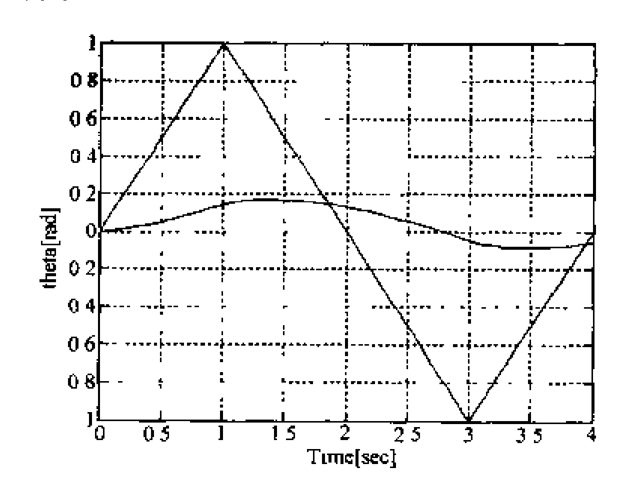


图 5.7　例 5-3 的响应曲线

【例 5-4】 反馈系统如图 5.5 所示，其中 $G(s)=\dfrac{s+2}{s^2+10s+1}$，系统输入信号为如图 5.8 所示的线齿波，试用 Simulink 求取系统输出响应，并将输入输出信号对比显示。

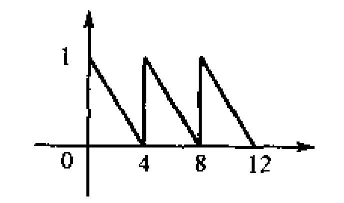


图 5.8　例 5-4 输入曲线

解：使用 Simulink 可以方便地实现对系统的建模和仿真，Simuliuk 的模型如图 5.9 所示。

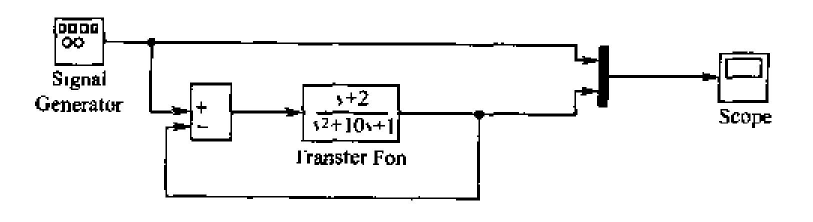


图 5.9　例 5-4 的 Simulink 模型

图 5.9 中，"Transfer Fcn" 模块建立 $G(s)$ 的模型，"Signal Generator" 产生线入的锯齿波，其参数设置如图 5.10 所示。按照题意，设置时在 "Wave form" 中选中 "sawtooth"

（锯齿波），在“Amplitude”（幅值）中输入 1，在“Frequency”中输入 1/4，在“Units”（单位）选中“Hertz”（赫兹）。Simulink 中还提供了其他类型的信号发生器，其使用方法和本题类似，这里不做详细介绍。

模型连好后进行仿真，仿真结束后双击示波器，输出图形如图 5.11 所示。

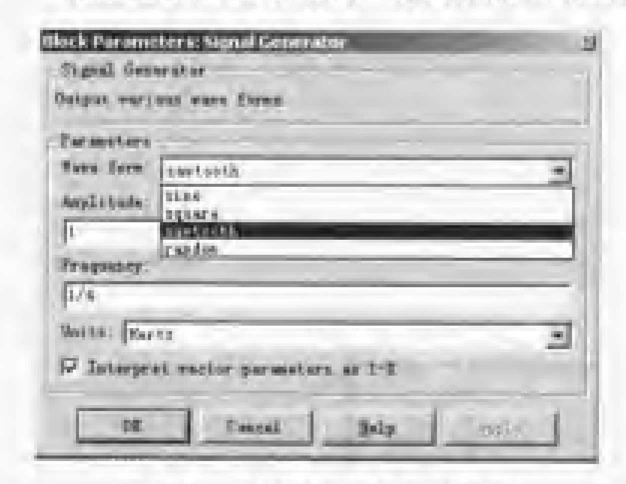


图 5.10　信号发生器参数设置界面

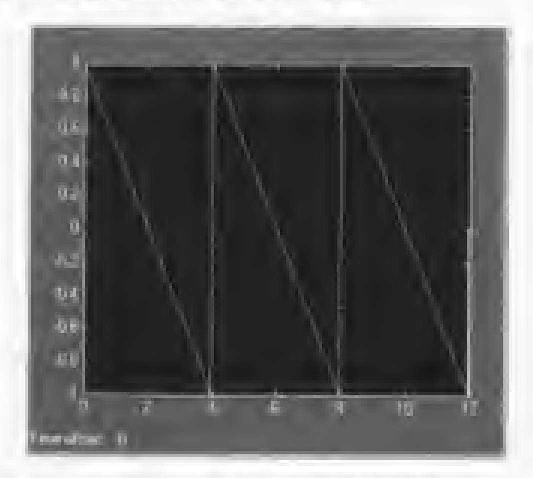


图 5.11　例 5-4 输出和输入曲线

【例 5-5】　反馈系统如图 5.12 所示，其中 $G_1(s)=\dfrac{s+5}{(s+1)(s+3)}$，$G_2(s)=\dfrac{s^2+1}{s^2+4s+4}$，系统输入信号为 $r(t)=\sin(t)$，试用 Simulink 求取系统输出响应，并将输入和输出信号对比显示。

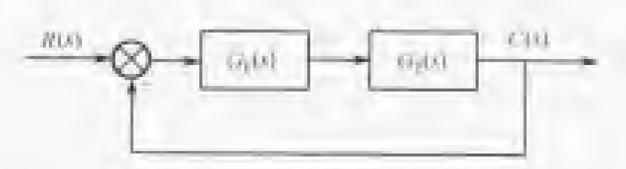


图 5.12　例 5-5 系统框图

解：Simulink 的模型如图 5.13 所示。

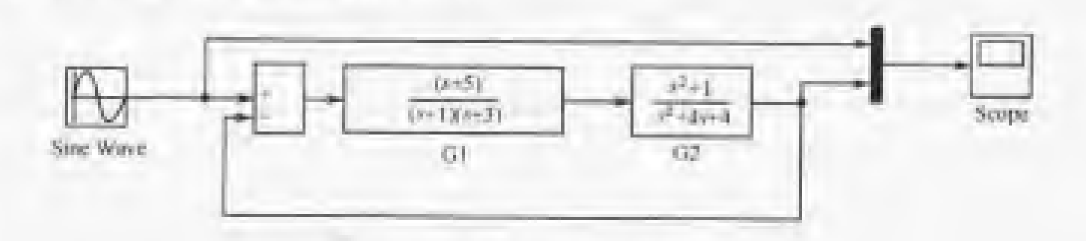


图 5.13　例 5-5 的 Simulink 模型

Simulink 中提供了用来建立系统模型的传递函数模块（Transfer Fcn）和零极点模块（Zero-Pole），可以方便地实现对系统的建模。本例中，$G_1(s)$ 是用零极点表示的，选用“Zero-Pole”非常方便；$G_2(s)$ 是用传递函数表示的，选用“Transfer Fcn”非常方便。信号源选择“Sine Wave”即可。

模型连好后进行仿真，仿真结束后，双击示波器，输出图形如图 5.14 所示。

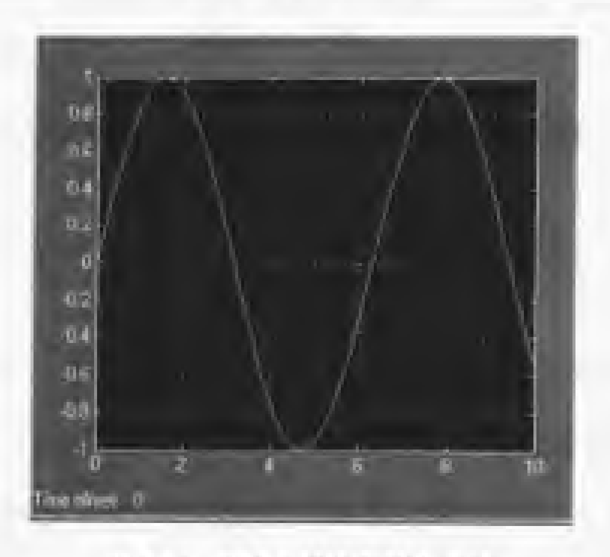


图 5.14 例 5-5 输出和输入曲线

5.2.5 一阶和二阶系统时域响应分析

5.2.5.1 一阶系统分析

可以用一阶微分方程描述的系统称为一阶系统，其传递函数为：

$$G(s)=\frac{K}{Ts+1} \tag{5-26}$$

其中，T 称为一阶系统的时间常数，$G(s)$ 可写成：

$$G(s)=\frac{C(s)}{R(s)} \tag{5-27}$$

当 $r(t)=1(t)$ 时，一阶系统的输出 $c(t)$ 称为单位阶跃响应。

此时，

$$R(s)=\frac{1}{s} \tag{5-28}$$

则

$$C(s)=G(s)R(s)=\frac{K}{s(Ts+1)}=K(\frac{1}{s}-\frac{1}{s+\frac{1}{T}}) \tag{5-29}$$

对式（5-29）进行拉氏反变换，得

$$c(t)=K(1-e^{-\frac{t}{T}}) \tag{5-30}$$

$c(t)$ 的波形如图 5.15 所示。

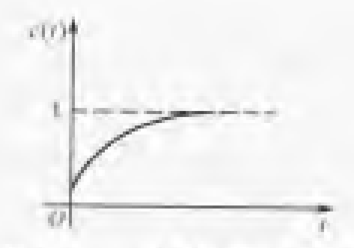


图 5.15 一阶系统的单位阶跃响应曲线

一阶系统时域响应的性能指标如下。

（1）调整时间 t_s：经过时间 $3T$～$4T$，响应曲线已达稳态值的 95%～98%，可以认为其调整过程已完成，故一般取 $t_s=(3\sim4)T$。

$$t_s=\begin{cases}3T,\Delta=0.05\\4T,\Delta=0.02\end{cases}$$

（2）稳态误差 e_{ss}：系统的实际输出 $c(t)$ 在时间 t 趋于无穷大时将趋近输入值，即

$$e_{ss}=\lim_{t\to\infty}[c(t)-r(t)]=0$$

（3）超调量 $\sigma_p\%$：一阶系统的单位阶跃响应为非周期响应，是单调的，故系统无振荡、无超调，$\sigma_p\%=0$。

一阶系统的闭环极点 $-p=\frac{1}{T}$，位于实轴上，由此可得到以下两条结论：

- 如果系统闭环极点位于负实轴上，则阶跃响应是单调的，$\sigma_p\%=0$。
- 调节时间 $t_s=\frac{3}{p}$，即闭环极点离虚轴距离越远则响应越快。

5.2.5.2 二阶系统分析

典型二阶系统的结构如图 5.16 所示，其闭环传递函数 $G(s)$ 为：

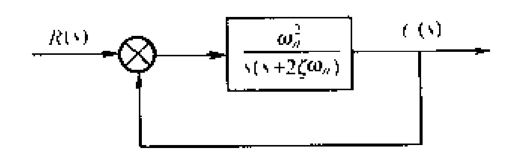


图 5.16 典型二阶系统的结构图

$$G(s)=\frac{\omega_n^{\ 2}}{s^2+2\zeta\omega_n s+\omega_n^{\ 2}} \tag{5-31}$$

或

$$\frac{C(s)}{R(s)}=\frac{1}{T^2s^2+2\zeta Ts+1} \tag{5-32}$$

式中，ω_n 为无阻尼自由振荡角频率，简称固有频率；ζ 为阻尼系数；$T=\frac{1}{\omega_n}$ 为系统振荡周期。

系统的特征方程为：

$$D(s)=s^2+2\zeta\omega_n s+\omega_n^2=0 \tag{5-33}$$

特征根为：

$$s_{1,2}=-\zeta\omega_n\pm\omega_n\sqrt{\zeta^2-1} \tag{5-34}$$

因此，二阶系统的两个极点为：

$$-p_{1,2}=-\zeta\omega_n\pm\omega_n\sqrt{\zeta^2-1} \tag{5-35}$$

在不同阻尼比下两个极点有不同的特征，因此其时域响应特征也不同。

1. 零阻尼（$\zeta=0$）

此时两个极点是一对纯虚根，$-p_{1,2}=\pm j\omega_n$，可求得其单位阶跃响应为：

$$c(t) = 1 - \cos(\omega_n t) \tag{5-36}$$

单位阶跃响应曲线如图 5.17 所示，是一种等幅振荡曲线，振荡角频率就是 ω_n。

2. 欠阻尼$(0<\zeta<1)$

此时两个极点是一对负实部的共轭复根，$-p_{1,2} = -\zeta\omega_n \pm \mathrm{j}\omega_n\sqrt{1-\zeta^2}$，其单位阶跃响应如图 5.18 所示，是一种衰减振荡曲线。

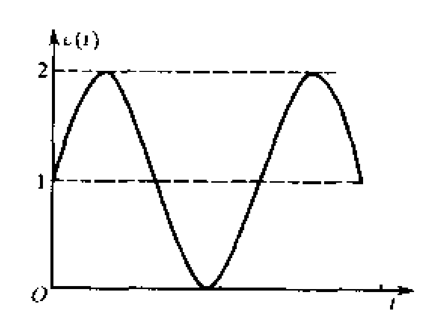


图 5.17　零阻尼系统单位阶跃响应曲线

图 5.18　欠阻尼系统单位阶跃响应曲线

曲线的表达式可表示为：

$$c(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} \mathrm{e}^{-\zeta\omega_n t} \sin(\omega_n\sqrt{1-\zeta^2}\,t + tg^{-1}\frac{1-\zeta}{\zeta}) \tag{5-37}$$

通常可设 $\sigma = \zeta\omega_n$ 为衰减指数；$\omega_d = \omega_n\sqrt{1-\zeta^2}$ 为振荡角频率；$\theta = tg^{-1}\frac{1-\zeta}{\zeta}$ 为初相角。

其他性能指标可用以下方法求取。

（1）上升时间 t_r

由

$$c(t)|_{t=t_r} = 1$$

即

$$\sin(\omega_d t_r + \theta) = 0$$

$$\omega_d t_r + \theta = n\pi, n = 0,1,\ldots$$

由于第一次达到稳态值的时间取 $n=1$，则

$$t_r = \frac{\pi - \theta}{\omega_d} \tag{5-38}$$

（2）峰值时间 t_p

令

$$\left.\frac{\mathrm{d}c(t)}{\mathrm{d}t}\right|_{t=t_p} = 0$$

可得

$$t_p = \frac{n\pi}{\omega_d}$$

由于第一次达到的峰值取 n=1，则

$$t_p = \frac{\pi}{\omega_d}$$

（3）超调量$\sigma_p\%$

由

$$\sigma_p\% = \frac{c(t_p) - c(\infty)}{c(\infty)} \times 100\% = \frac{c(t_p) - 1}{1} \times 100\%$$

可得

$$\sigma_p\% = \mathrm{e}^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} \times 100\% = \mathrm{e}^{-\frac{\pi}{\mathrm{tg}\theta}} \times 100\% \tag{5-39}$$

（4）超调时间t_s

按定义，

$$\left|\frac{c(t) - c(\infty)}{c(\infty)}\right| \leqslant \Delta$$

即

$$\left|\frac{1}{\sqrt{1-\zeta^2}} \mathrm{e}^{-\zeta\omega_n t} \sin(\omega_n \sqrt{1-\zeta^2} t + \mathrm{tg}^{-1} \frac{1-\zeta}{\zeta})\right| \leqslant \Delta$$

由于正弦项的绝对值总小于 1，故上式可近似地表示为：

$$\left|\frac{1}{\sqrt{1-\zeta^2}} \mathrm{e}^{-\zeta\omega_n t}\right| \leqslant \Delta$$

即

$$\frac{1}{\sqrt{1-\zeta^2}} \mathrm{e}^{-\zeta\omega_n t} \leqslant \Delta$$

由于该式是单调下降的，取等号即可，故经化简可得：

$$t_s = \frac{\ln\frac{1}{\Delta} + \ln\frac{1}{\sqrt{1-\zeta^2}}}{\zeta\omega_n} \tag{5-40}$$

在常用的ζ范围（0.4～0.9）内，$\ln\frac{1}{\sqrt{1-\zeta^2}} = 0.087 \sim 0.8$，平均取 0.5 是合适的，故

$$t_s = \begin{cases} \dfrac{3.5}{\zeta\omega_n}, \Delta = 0.05 \\ \dfrac{4.5}{\zeta\omega_n}, \Delta = 0.02 \end{cases} \tag{5-41}$$

3. 临界阻尼($\zeta = 1$)

此时两个极点是一对负实数重极点，$-p_{1,2} = -\omega_n$，其单位阶跃响应表达式可表示为：

$$c(t)=1-e^{-\omega_n t}(1+\omega_n t) \tag{5-42}$$

其单位阶跃响应如图 5.19 所示。

由图可见，$\zeta=1$时，阶跃响应正好进入单调无超调状态($\sigma_p\%=0$)，故可从这个意义上定义其临界。临界阻尼下的调节时间可以通过数值计算来获得。

$$t_s=\frac{4.5}{\omega_n},\Delta=0.05 \tag{5-43}$$

4. 过阻尼（$\zeta>1$）

此时两个极点是两个不相等的负实数极点，$-p_{1,2}=-\zeta\omega_n\pm\omega_n\sqrt{\zeta^2-1}$

令

$$T_1=-\frac{1}{-p_1}=\frac{1}{\zeta\omega_n-\omega_n\sqrt{\zeta^2-1}}$$

$$T_2=-\frac{1}{-p_2}=\frac{1}{\zeta\omega_n+\omega_n\sqrt{\zeta^2-1}}$$

则

$$G(s)=\frac{\omega_n^2}{(s+p_1)(s+p_2)}=\frac{1}{(T_1s+1)(T_2s+1)},\ T_1>T_2 \tag{5-44}$$

其单位阶跃响应表达式可表示为：

$$c(t)=1+\frac{T_2}{T_1-T_2}e^{-\frac{t}{T_2}}+\frac{T_1}{T_2-T_1}e^{-\frac{t}{T_1}} \tag{5-45}$$

响应曲线如图 5.20 所示。

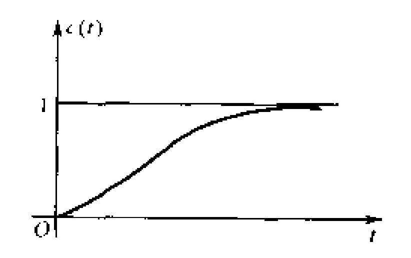


图 5.19　临界阻尼系统单位阶跃响应曲线

图 5.20　过阻尼系统单位阶跃响应曲线

从图可以看出，响应仍是一个单调过程，$\sigma_p\%=0$，其调节时间t_s可通过数值计算来确定，ζ越大，即T_1和T_2越错开，t_s越大。

从图中可以看出一个重要现象，即当$T_1>>T_2$时，对响应表达式中的两个分量，只有第二分量（与T_1对应）起主要作用，而第一分量（与T_2对应）仅仅影响时域响应的起始点。一般认为，当$T_1>>5T_2$时，T_2的影响就可以忽略不计了，即

$$c(t)\approx 1-e^{-\frac{t}{T_1}} \tag{5-46}$$

相应地

$$G(s)=\frac{1}{(T_1s+1)(T_2s+1)}\approx\frac{1}{T_1s+1} \tag{5-47}$$

此时二阶系统就可以近似地作为一阶系统来分析了。

5.2.5.3 二阶系统参数对时域响应性能的影响

1. 闭环参数 ω_n 和 ζ 的影响

从上面对性能指标的分析可知，t_r，t_p 和 t_s 均与 ω_n 成反比，因此从对快速性的影响而言，ω_n 越大则响应越快。当然，ζ 在一定程度上也对快速性有影响。一般而言，ζ 越小，快速性能越好，但由于 ζ 在实际中允许变化的范围是有限的，因此其对系统快速性的影响也是有限的。

另一方面，ζ 惟一决定了 $\sigma_p\%$ 的大小，也就是说，ζ 是决定系统相对稳定性的惟一因素，ζ 越大，$\sigma_p\%$ 越小。

【例 5-5】 已知一个如图 5.21 所示的典型二阶系统，其开环传递函数为 $G(s)=\frac{\omega_n^{\ 2}}{s(s+2\zeta\omega_n)}$，其中 $\omega_n=1$，ζ 为阻尼比，试绘制 ζ 分别为 0, 0.2, 0.4, 0.6, 0.9, 1.2, 1.5 时其单位负反馈系统的单位阶跃响应曲线（绘制在同一张图上）。

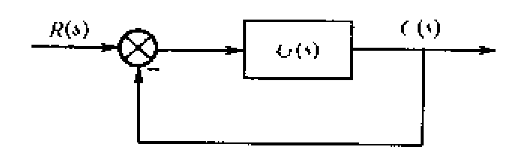


图 5.21 例 5-6 系统框图

解：MATLAB 程序代码如下。

```
wn=1
sigma=[0, 0.2, 0.4, 0.6, 0.9, 1.2, 1.5]
num=wn*wn
t=linspace(0, 20, 200)'
for j=1:7
den=conv([1, 0], [1, 2*wn*sigma(j)]);
s1=tf(num, den)
sys=feedback(s1, 1)
y(:, j)=step(sys, t);
end
plot(t, y(:, 1:7))
grid
gtext('sigma=0')
```

```
gtext('sigma=0.2')
gtext('sigma=0.4')
gtext('sigma=0.6')
gtext('sigma=0.9')
gtext('sigma=1.2')
gtext('sigma=1.5')
```

程序输出如图 5.22 所示。

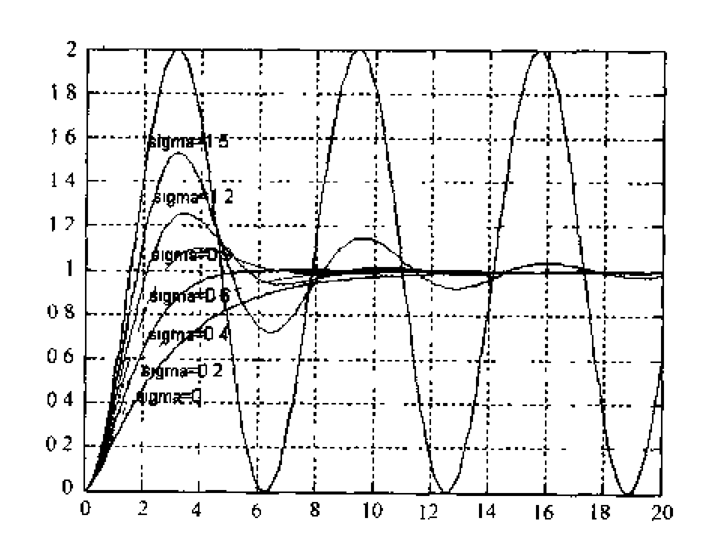


图 5.22　例 5-6 输出结果

从图中可以明显看出，在 ζ 为 0.4～0.9 的范围内，系统上升较快，而超调又不太大，故在一般工程系统中，ζ 就选在这个范围内，其中尤以 $\zeta=\frac{\sqrt{2}}{2}$ 时响应较快，而此时 $\sigma_p\%$ 仅为 4.3%，通常将此称为最佳阻尼，而具有最佳阻尼的二阶系统就称为二阶最佳系统。

2. 开环参数 *K* 和 *T* 的影响

对一般的二阶系统而言，通过适当的变换，其闭环传递函数可用式（5-48）表示，其中 *K* 为回路增益，通常是可调节的，*T* 为时间常数，通常由受控对象的特性决定，一般是不可以改变的。

$$G(s)=\frac{C(s)}{R(s)}=\frac{\frac{K}{T}}{s^2+\frac{1}{T}s+\frac{K}{T}} \tag{5-48}$$

对比二阶系统的典型传递函数，可设 $\frac{K}{T}=\omega_n{}^2$，$\frac{1}{T}=2\zeta\omega_n$，即

$$\omega_n=\sqrt{\frac{K}{T}} \tag{5-49}$$

$$\zeta = \frac{1}{2\sqrt{KT}} \tag{5-50}$$

可见 T 越小，则 ω_n 越大，ζ 也越大，系统的快速性和相对稳定性同时转好。但在实际系统中，用 T 来改善系统性能的作用是有限的。

另一方面，K 越大，则 ω_n 越大，而 ζ 越小，表明 K 对快速性和相对稳定性的影响是矛盾的。在实际系统中，应根据系统的要求适当折中。

对二阶最佳系统而言，应有 $K = \frac{1}{2T}$，称之为二阶最佳参数关系。

【例 5-7】 已知一个如图 5.23 所示的二阶系统，其开环传递函数 $G(s) = \frac{k}{s(Ts+1)}$，其中 $T=1$，试绘制 k 分别为 0.1, 0.2, 0.5, 0.8, 1.0, 2.4 时其单位负反馈系统的单位阶跃响应曲线（绘制在同一张图上）。

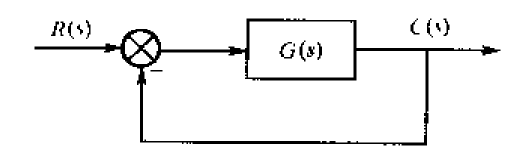


图 5.23 例 5-7 系统框图

解：MATLAB 程序代码如下。

```
T=1
k=[0.1, 0.2, 0.5, 0.8, 1.0, 2.4]
t=linspace(0, 20, 200)'
num=1
den=conv([1, 0], [T, 1])
for j=1:6
s1=tf(num*k(j), den)
sys=feedback(s1, 1)
y(:, j)=step(sys, t);
end
plot(t, y(:, 1:6))
grid
gtext('k=0.1')
gtext('k=0.2')
gtext('k=0.5')
gtext('k=0.8')
gtext('k=1.0')
gtext('k=2.4')
```

程序输出如图 5.24 所示。

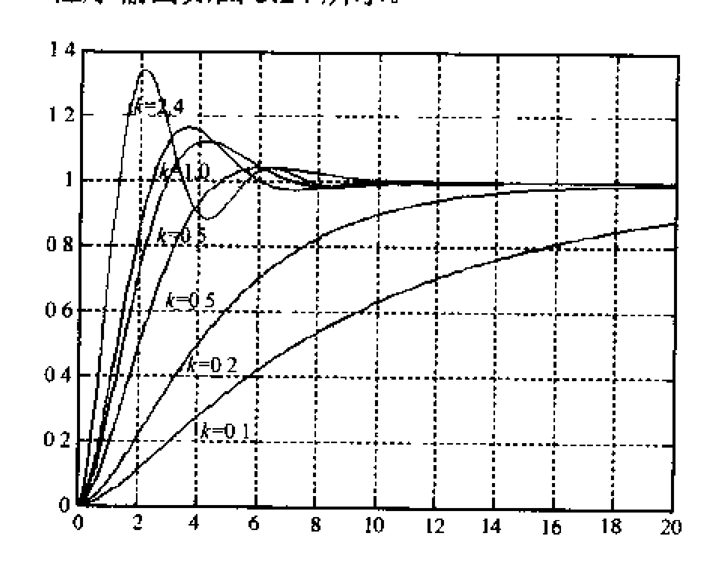


图 5.24 例 5-7 输出结果

5.2.5.4 闭环极点分布对时域响应的影响

闭环极点分布对时域响应的影响可归结为以下几点：

（1）如果闭环极点落于虚轴上，则系统处于临界稳定状态。

（2）如果闭环极点是负实数极点，则系统阶跃响应是单调的，$\sigma_p\%=0$。

（3）如果闭环极点是负实部的共轭复数极点，则系统阶跃响应是衰减振荡的，其超调量与初相角θ有关，θ角越大，则超调量越大。

（4）系统时域响应的快速性与闭环极点距虚轴的距离有关，距离越大，则t_s越小。

（5）如果系统有多个闭环极点，则距虚轴越近的闭环极点所起的作用越大，如果一个闭环极点距虚轴的距离较另一个闭环极点距虚轴的距离大 5 倍或 5 倍以上，则距离远的闭环极点的影响可以忽略不计。

5.2.5.5　改善系统时域响应性能的一些措施

如前文所述，想单纯通过改变回路增益 K 来达到系统时域响应又快又稳是不可能的，只能做一个合理的折中。下面给出两种可以提高系统性能的方法。

1. 输出微分反馈

在基本系统的基础上，从输出向输入端引入附加的微分负反馈控制，如图 5.25 所示。

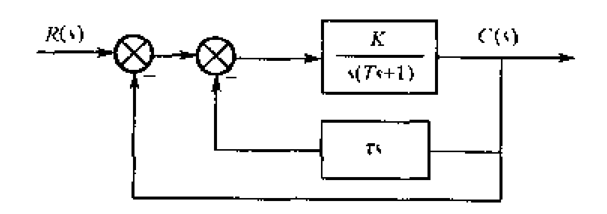


图 5.25　带输出微分反馈的二阶系统

可按 K、T 与 ω_n、ζ 的关系，将原开环传递函数改写为：

$$\frac{K}{s(Ts+1)}=\frac{\omega_n^{\ 2}}{s(s+2\zeta\omega_n)} \tag{5-51}$$

式中，ω_n 和ζ是$\tau=0$ 时原系统的固有频率和阻尼系数。当τ不等于 0 时，由其闭环传递函数可推出：

$$G(s)=\frac{C(s)}{R(s)}=\frac{\omega_n^{\ 2}}{s^2+2(\zeta+\frac{1}{2}\tau\omega_n)s+\omega_n^{\ 2}} \tag{5-52}$$

显然，加微分反馈后，系统的固有频率 ω_n 不变，而阻尼比提离。

$$\zeta^{'}=\zeta+\frac{1}{2}\tau\omega_n \tag{5-53}$$

由于输出锁分反馈可以在不改变快速性的条件下提高相对稳定性，因此实际中可通过提高 K 来进一步提高快速性，而用τ 来保证必要的相对稳定性，即采用输出反馈，这样既可以提高系统的相对稳定性，又可以提高其快速性。

【例 5-8】　已知一个如图 5.23 所示的二阶系统，其开环传递函数$G(s)=\dfrac{k}{s(Ts+1)}$，其中$T=1$，试绘制k分别为 0.1, 0.2, 0.5, 0.8, 1.0, 2.4 时其单位负反馈系统的单位阶跃响应

曲线（绘制在同一张图上）。

解：MATLAB 程序代码如下。

```
T=1
k=[0.1, 0.2, 0.5, 0.8, 1.0, 2.4]
t=linspace(0, 20, 200)'
num=1
den=conv([1, 0], [T, 1])
for j=1:6
s1=tf(num*k(j), den)
sys=feedback(s1, 1)
y(:, j)=step(sys, t);
end
plot(t, y(:, 1:6))
grid
gtext('k=0.1')
gtext('k=0.2')
gtext('k=0.5')
gtext('k=0.8')
gtext('k=1.0')
gtext('k=2.4')
```

程序输出如图 5.26 所示。

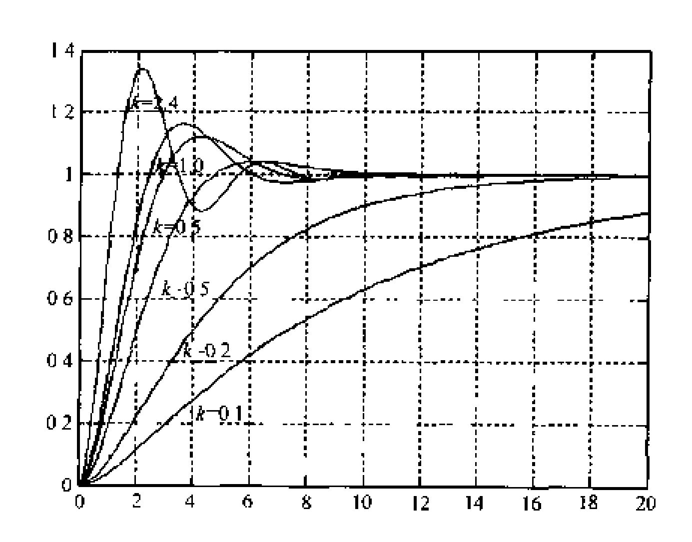


图 5.26　例 5-8 输出结果

2. 比例微分控制

在基本系统的基础上，在前向通道中增加微分控制就构成了带比例微分控制的二阶系统，如图 5.27 所示。

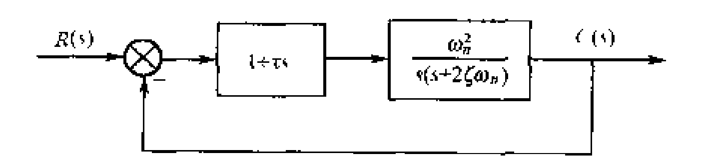


图 5.27　带比例微分控制的二阶系统

其闭环传递函数为：

$$G(s)=\frac{C(s)}{R(s)}=\frac{(1+\tau s)\omega_n^{\ 2}}{s^2+2(\zeta+\frac{1}{2}\tau\omega_n)s+\omega_n^{\ 2}} \tag{5-54}$$

由此可知，比例微分控制同样能实现在不改变 ω_n 的条件下提高系统阻尼比 $\zeta'=\zeta+\frac{1}{2}\tau\omega_n$ 的效果，作用类似输出微分反馈控制。但与输出微分反馈控制不同的是，在闭环传递函数中增加了一个零点 $z=-\frac{1}{\tau}$，分析表明，它的存在将使系统的上升加快，

但$\sigma_p\%$会有所增加，其趋势随τ的加大而加大。

【例 5-9】　设系统闭环传递函数为：

$$G(s)=\frac{4(1+\tau s)}{s^2+2s+4}$$

试求取$\tau=0$, 0.2, 0.4 时的单位阶跃响应（绘制在同一张图上）。

解：MATLAB 程序代码如下。　　程序输出如图 5.28 所示。

```
tou=[0, 0.2, 0.4]
t=linspace(0, 8, 80)'
num=4
den=[1, 2, 4]
for j=1:3
sys=tf( conv(num, [tou(j), 1]), den)
y(:, j)=step(sys, t);
end
plot(t, y(:, 1:3))
grid
gtext('tou=0')
gtext('tou=0.2')
gtext('tou=0.4')
```

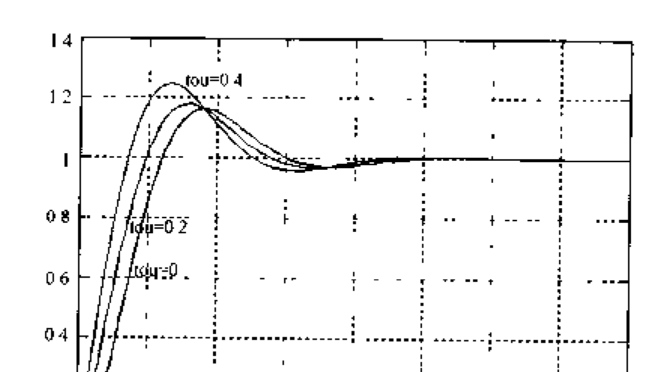


图 5.28　例 5-9 输出结果

5.2.6　高阶系统的时域分析

5.2.6.1　高阶系统时域响应的一般形式

设系统的闭环传递函数为：

$$\begin{aligned}G(s)=\frac{C(s)}{R(s)}&=\frac{b_0s^m+b_1s^{m-1}+\cdots+b_{m-1}s+b_m}{a_0s^n+a_1s^{n-1}+\cdots+a_{n-1}s+a_n}\\&=\frac{K(s+z_1)(s+z_2)\cdots(s+z_m)}{(s+p_1)(s+p_2)\cdots(s+p_n)}\end{aligned}\tag{5-55}$$

当输入$r(t)=1(t)$时，$R(s)=\dfrac{1}{s}$，此时

$$C(s)=G(s)R(s)=\frac{K(s+z_1)(s+z_2)\cdots(s+z_m)}{(s+p_1)(s+p_2)\cdots(s+p_n)}\cdot\frac{1}{s}\tag{5-56}$$

为使问题简单化一些，可设$G(s)$中无重极点，则

$$C(s)=\frac{K\prod_{j=1}^{m}(s-z_j)}{\prod_{i=1}^{n}(s-p_i)}\cdot\frac{1}{s}=\frac{A_0}{s}+\sum_{i=1}^{n}\frac{A_i}{(s-p_i)}\tag{5-57}$$

其中，

$$A_i = \lim_{s \to -p_i} C(s)(s+p_i)\text{，}\quad i = 0,1,\cdots,n, -p_0 = 0 \tag{5-58}$$

则

$$c(t) = A_0 + \sum_{i=1}^{n} A_i e^{-p_i t} \tag{5-59}$$

一般地，若 $G(s)$ 的极点中有 q 个负实数极点、r 个负实部共轭复数极点，则上式还可改写为：

$$C(s) = \frac{K\prod_{j=1}^{m}(s+z_j)}{s\prod_{i=1}^{n}(s+p_i)\prod_{k=1}^{r}(s^2+2\zeta_k\omega_k s+\omega_k^2)} \tag{5-60}$$

$$= \frac{A_0}{s} + \sum_{i=1}^{q}\frac{A_i}{s+p_i} + \sum_{k=1}^{r}\frac{B_k(s+\zeta_k\omega_k)+C_k\omega_k\sqrt{1-\zeta_k^2}}{s^2+2\zeta_k\omega_k s+\omega_k^2}$$

其中，$r+2q=m$，则

$$c(t) = A_0 + \sum_{i=1}^{q}A_i e^{-p_i t} + \sum_{k=1}^{r}B_k e^{-\zeta_k\omega_k t}\cos\omega_k\sqrt{1-\zeta_k^2}\,t + \sum_{k=1}^{r}C_k e^{-\zeta_k\omega_k t}\sin\omega_k\sqrt{1-\zeta_k^2}\,t \tag{5-61}$$

5.2.6.2　高阶系统的主导极点

由式（5-59）可知，高阶系统阶跃响应是由一系列动态分量组成的，各动态分量的幅值由闭环极点和零点共同决定。

由式（5-58）可知，当某个极点与某个零点接近时，其幅值必定很小，其动态分量的衰减速度由其极点的实部，即闭环极点距虚轴的距离决定。距虚轴越远的闭环极点，其所对应的动态分量衰减越快。显然，在阶跃响应过程中，影响最大的分量是那些幅值最大而衰减又最慢的分量，这些分量所对应的闭环极点是那些距虚轴最近而附近又没有闭环零点的闭环极点。

由此可得以下结论：

（1）主导极点。在整个响应过程中，起决定性作用的是闭环极点，称之为主导极点，它是距虚轴最近而附近又没有闭环零点的闭环极点。工程上往往只用主导极点来估算系统的动态特性，即将系统近似地看成是一阶或二阶系统。

（2）距虚轴的距离较主导极点远 5 倍或 5 倍以上的闭环零点、极点，其影响可以忽略不计。

（3）偶极子。一对靠得很近的闭环零点、极点称为偶极子。工程上，当某极点与某零点之间的距离比它们的模值小一个数量级时，就可认为这对零点、极点为偶极子。偶极子对时域的影响可以忽略不计。在闭环传递函数中，如果零点、极点数值上相近，则可将该零点和极点一起消掉，称为偶极子相消。

（4）除主导极点外，闭环零点的作用是使响应加快而超调增加，闭环极点的作用则正好相反。

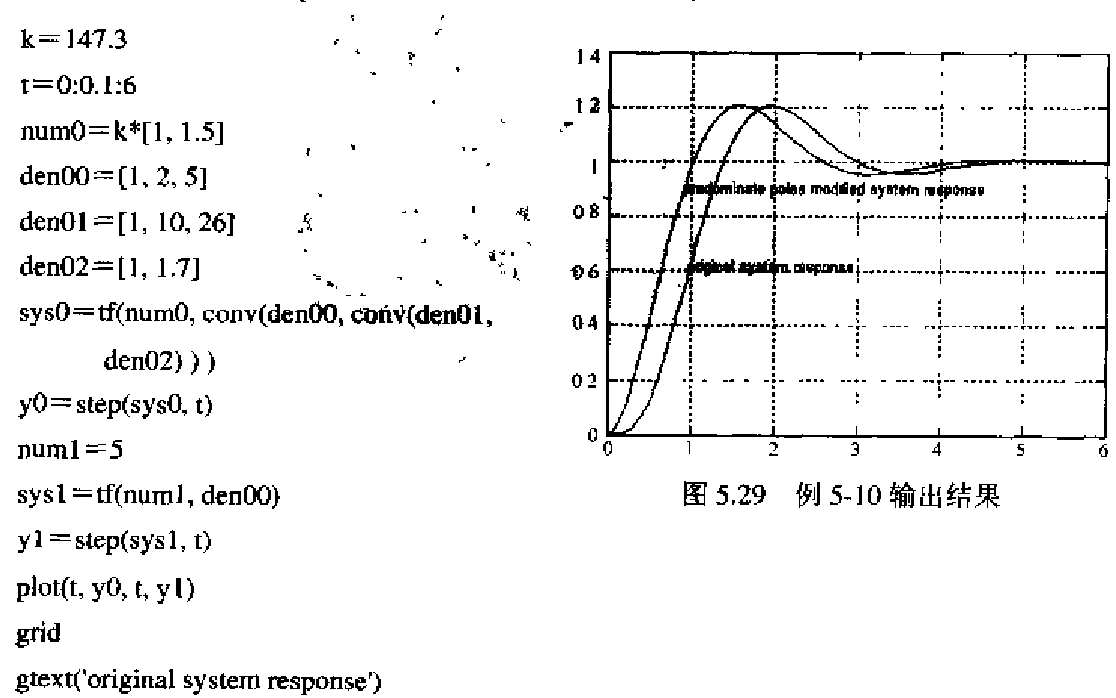
【例 5-10】　设系统的传递函数 $G(s)$ 为：

$$G(s)=\frac{147.3(s+1.5)}{(s^2+2s+5)(s^2+10s+26)(s+1.7)}$$

试分析其主导极点，并比较由主导极点构成的系统与原系统的单位阶跃响应。

解：系统有 5 个极点：$p_{1,2}=-5\pm\text{i}$，$p_{3,4}=-1\pm2\text{i}$，$p_5=-1.7$ 和一个零点 $z_1=-1.5$。显然，主导极点为 $p_{3,4}=-1\pm2\text{i}$，由主导极点构成的系统传递函数为 $G(s)=\dfrac{5}{(s^2+2s+5)}$，值得注意的是，两个传递函数的静态增益应该相同。

计算阶跃响应的 MATLAB 代码如下：　　　　　　输出结果如图 5.29 所示。




```
k=147.3
t=0:0.1:6
num0=k*[1, 1.5]
den00=[1, 2, 5]
den01=[1, 10, 26]
den02=[1, 1.7]
sys0=tf(num0, conv(den00, conv(den01,
        den02)))
y0=step(sys0, t)
num1=5
sys1=tf(num1, den00)
y1=step(sys1, t)
plot(t, y0, t, y1)
grid
gtext('original system response')
gtext('predominate poles modified system response')
```

图 5.29　例 5-10 输出结果

从图 5.29 可以看出，主导极点构成的系统和原系统在动态性能上差别很小。

【例 5-11】　设系统的闭环传递函数 $G(s)$ 为：

$$G(s)=\frac{500}{(s^2+10s+50)(s+10)}$$

试分析其主导极点，并比较由主导极点构成的系统与原系统的单位阶跃响应。

解：系统有 3 个极点 $p_{1,2}=-5\pm5\text{i}$，$p_3=-10$。显然，主导极点为 $p_{1,2}=-5\pm5\text{i}$，由主导极点构成的系统传递函数为 $G'(s)=\dfrac{50}{(s^2+10s+50)}$，值得注意的是，两个传递函数的静态增益应该相同。

Simulink 的模型如图 5.30 所示。图中 “Transfer Fcn” 建立主导极点构成系统的部分，“Transfer Fcn1” 建立非主导极点构成系统的部分，这两部分串联。

模型连好后，进行仿真，仿真结束后，双击示波器，输出图形如图 5.31 所示。

从图 5.31 可以看出，主导极点构成的系统和原系统在动态性能上差别很小。

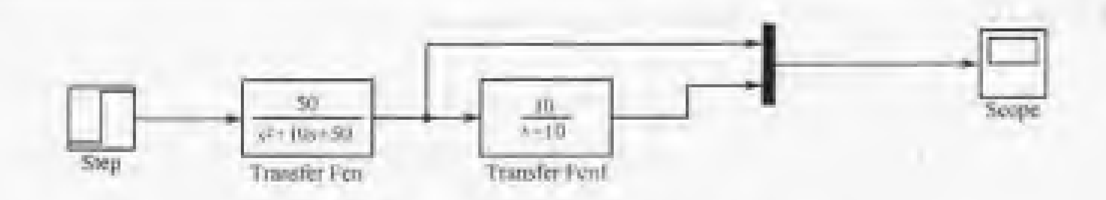


图 5.30　例 5-11 的 Simulink 模型

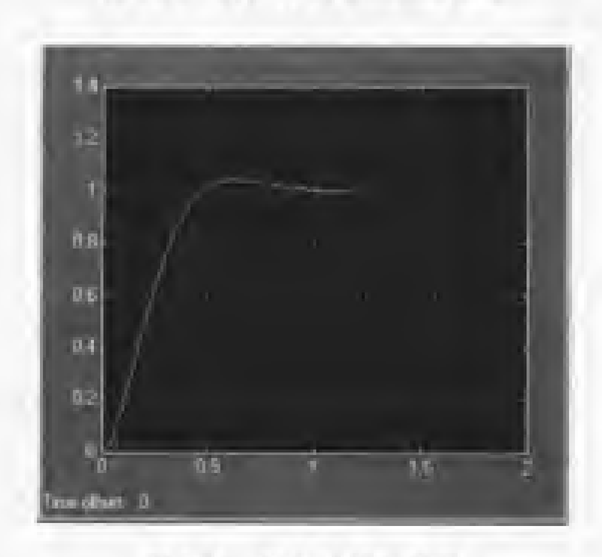


图 5.31　例 5-11 输出结果

5.3　稳定性分析

自动控制系统能够正常工作的首要条件是系统必须是稳定的。稳定性的问题也是控制理论发展过程中首先涉及到的一个具有实际意义的理论问题。

稳定性的提法有多种，此书仅介绍在工程中常用的经典提法。若控制系统在足够小的初始偏差作用下，其过渡过程随时间的推移逐渐衰减并趋于零，即具有恢复原平衡状态的能力，则称该系统是稳定的，否则，称该系统是不稳定的。

5.3.1　稳定性基本概念

系统稳定就是要求系统时域响应的动态分量随时间的推移而最终趋于零。

设 n 阶线性定常系统的微分方程为：

$$\begin{aligned}&a_0\frac{\mathrm{d}^n c(t)}{\mathrm{d}t^n}+a_1\frac{\mathrm{d}^{n-1}c(t)}{\mathrm{d}t^{n-1}}+\cdots+a_{n-1}\frac{\mathrm{d}c(t)}{\mathrm{d}t}+a_n c(t)\\&=b_0\frac{\mathrm{d}^m r(t)}{\mathrm{d}t^m}+b_1\frac{\mathrm{d}^{m-1}r(t)}{\mathrm{d}t^{m-1}}+\cdots+b_{m-1}\frac{\mathrm{d}r(t)}{\mathrm{d}t}+b_m r(t)\end{aligned}\tag{5-62}$$

对式（5-62）进行拉氏变换，得

$$C(s)=\frac{M(s)}{D(s)}R(s)+\frac{N(s)}{D(s)}\tag{5-63}$$

在式（5-63）中取 $R(s)=0$，得到在初始状态影响下系统的时间响应（即零输入响应）为：

$$C(s)=\frac{N(s)}{D(s)} \tag{5-64}$$

若 p_i 为系统特征方程 $D(s)=0$ 的根，当 p_i 各不相同时，有：

$$c(t)=L^{-1}[C(s)]=L^{-1}\left[\frac{N(s)}{D(s)}\right]=\sum_{i=1}^{n}A_i\mathrm{e}^{p_it} \tag{5-65}$$

若系统所有特征根 p_i 的实部均为负值，即 $\mathrm{Re}[p_i]<0$，则零输入响应（暂态响应）最终将衰减到零，即

$$\lim_{t\to\infty}c(t)=0 \tag{5-66}$$

这样的系统就是稳定的。

反之，若特征根中有一个或多个根具有正实部，则暂态响应将随时间的推移而发散，即

$$\lim_{t\to\infty}c(t)=\infty \tag{5-67}$$

这样的系统是不稳定的。

综上所述，系统稳定的充分必要条件是系统特征根的实部均小于零，即系统的特征根均在根平面的左半平面。

5.3.2　稳定性判据

由稳定性定义可知，稳定性问题可归结为系统闭环极点的求取，即闭环特征方程根的求取问题。由于高阶代数方程式的根一般没有解析解，故只能用数值方法来求取。其次，用直接求根法来分析系统的稳定性，不易给出系统结构和参数与稳定性的关系，对系统的综合问题帮助不大，因此需要寻求各种不直接求解代数方程式的方法来判断系统的稳定性，这就是所谓的稳定性判据。常见的稳定性判接有劳斯判据和赫尔维茨判据。

1. 劳斯判据

设闭环特征方程式为：

$$D(s)=a_0s^n+a_1s^{n-1}+a_2s^{n-2}+\cdots+a_{n-1}s\ +a_n=0,\ \ a_0>0 \tag{5-68}$$

系统稳定的必要条件是：方程中各次项系数均大于 0，即 $a_i>0, i=1,2,\ldots,n$。

设 n 阶系统的特征方程为：

$$\begin{aligned}D(s)&=a_0s^n+a_1s^{n-1}+a_2s^{n-2}+\ldots+a_{n-1}s\ +a_n\\&=a_0(s-p_1)(s-p_2)\ldots(s-p_n)\\&=0\end{aligned} \tag{5-69}$$

将上式的系数排成如下所示的行和列，即构成劳斯阵列（劳斯表）。

$$
\begin{array}{lllll}
s^n & a_0 & a_2 & a_4 & a_6 & \cdots \\
s^{n-1} & a_1 & a_3 & a_5 & a_7 & \cdots \\
s^{n-2} & b_1 & b_2 & b_3 & b_4 & \cdots \\
s^{n-3} & c_1 & c_2 & c_3 & c_4 & \cdots \\
\cdots & \cdots & \cdots & & & \\
s^2 & f_1 & f_2 & & & \\
s^1 & g_1 & & & & \\
s^0 & h_1 & & & &
\end{array}
$$

其中，

$$
\begin{gathered}
b_1 = \frac{\begin{vmatrix} a_0 & a_2 \\ a_1 & a_3 \end{vmatrix}}{-a_1}; b_2 = \frac{\begin{vmatrix} a_0 & a_4 \\ a_1 & a_5 \end{vmatrix}}{-a_1}; b_3 = \frac{\begin{vmatrix} a_0 & a_6 \\ a_1 & a_7 \end{vmatrix}}{-a_1}, \cdots \\
c_1 = \frac{\begin{vmatrix} a_1 & a_3 \\ b_1 & b_2 \end{vmatrix}}{-b_1}; c_2 = \frac{\begin{vmatrix} a_1 & a_5 \\ b_1 & b_3 \end{vmatrix}}{-b_1}; c_3 = \frac{\begin{vmatrix} a_1 & a_7 \\ b_1 & b_4 \end{vmatrix}}{-b_1}, \cdots
\end{gathered} \tag{5-70}
$$

劳斯判据给出了控制系统稳定的充分条件，即劳斯表中第一列所有元素均大于零。

在使用劳斯判据时，对劳斯表计算中的一些特殊情况应引起注意：

- 如果劳斯表中某一行的第一个元素为零，而该行其他元素并不为零，在计算下一行的第一个元素时，该元素必将趋于无穷大，以至使劳斯表的计算无法往下进行，则用一个有限小的数值ε来代替该行的第一个元素，然后按照通常方法计算。
- 如果劳斯表中某一行的元素全部为零，表示在S平面内存在一些大小相等而符号相反的实根或共轭虚根，系统是不稳定的，则建立辅助方程式。

2. 赫尔维茨判据

设系统的特征方程为：

$$a_0 s^n + a_1 s^{n-1} + a_2 s^{n-2} + \cdots + a_{n-1} s + a_n = 0 \tag{5-71}$$

以特征方程的各项系数组成如下行列式：

$$
\Delta = \begin{vmatrix}
a_1 & a_0 & 0 & 0 & 0 & 0 & \cdots \\
a_3 & a_2 & a_1 & a_0 & 0 & 0 & \cdots \\
a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & \cdots \\
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \\
\vdots & \vdots & \vdots & \vdots & \vdots & & a_n
\end{vmatrix} \tag{5-72}
$$

赫尔维茨判据指出，系统稳定的充分必要条件是：在$a_0 > 0$情况下，上述行列式的各阶主子式Δ_i均大于零，即

$$\Delta_1 = a_1 > 0$$

$$\Delta_2 = \begin{vmatrix} a_1 & a_0 \\ a_3 & a_2 \end{vmatrix} = a_1 a_2 - a_0 a_3 > 0$$

$$\Delta_3=\begin{vmatrix}a_1 & a_0 & 0\\ a_3 & a_2 & a_1\\ a_5 & a_4 & a_2\end{vmatrix}>0$$

$$\vdots \tag{5-73}$$

$$\Delta_n=\Delta>0$$

当且仅当由系统分母多项式构成的赫尔维茨矩阵为正定矩阵时，系统稳定。

【例 5-12】 已知系统特征方程为：

$$s^4+6s^3+12s^2+11s+6=0$$

试用劳斯判据判断其稳定性。

解：列出如下劳斯表

$$\begin{array}{llll}
s^4 & 1 & 12 & 6\\
s^3 & 6 & 11 & \\
s^2 & \dfrac{\begin{vmatrix}1 & 12\\ 6 & 11\end{vmatrix}}{-6}=\dfrac{61}{6} & 6 & \\
s^1 & \dfrac{455}{61} & & \\
s^0 & 6 & &
\end{array}$$

从表中可以看出，劳斯表中第一列元素大于零，因此系统是稳定的，即所有特征根均在 S 平面的左半平面。

【例 5-13】 已知系统特征方程为：

$$s^3+10s^2+16s+160=0$$

试用劳斯判据判断其稳定性。

解：列出如下劳斯表

$$\begin{array}{lll}
s^3 & 1 & 16\\
s^2 & 10 & 160\\
s^1 & 0 & 0\\
s^1 & 20 & 0\\
s^0 & 160 &
\end{array}\qquad \text{辅助多项式}P(s)=10s^2+160$$

劳斯表中第一列元素符号没有改变，系统没有右半平面的根，但由 $P(s)=0$ 可得：

$$10s^2+160=0$$

$$s_{1,2}=\pm \mathrm{j}4$$

系统有一对共轭虚根，系统处于临界稳定状态，但从工程角度来看，临界稳定属于不稳定，因此该系统是不稳定的。

【例 5-14】 已知系统特征方程为：

$$s^5+2s^4+3s^3+6s^2-4s-8=0$$

试用劳斯判据判断其稳定性。

解：列出如下劳斯表

$$
\begin{array}{llll}
s^5 & 1 & 3 & -4 \\
s^4 & 2 & 6 & -8 \quad P(s)=2s^4+6s^2-8 \\
s^3 & 8 & 12 & 0 \quad P'(s)=8s^3+12s \\
s^2 & 3 & -8 & \\
s^1 & 33.3 & & \\
s^0 & -8 & &
\end{array}
$$

劳斯表中第一列元素符号改变一次，系统不稳定，且有一个右半平面的根，由 $P(s)=0$ 得：

$$2s^4+6s^2-8=0$$

$$s_{1,2}=\pm 1 \qquad s_{3,4}=\pm \mathrm{j}2$$

【例 5-15】 已知系统特征方程为：

$$a_0 s^3+a_1 s^2+a_2 s+a_3=0 \qquad (a_0>0)$$

试用赫尔维茨判据求其稳定的充分必要条件。

解：列出如下行列式Δ

$$\Delta=\begin{vmatrix} a_1 & a_0 & 0 \\ a_3 & a_2 & a_1 \\ 0 & 0 & a_3 \end{vmatrix}$$

由赫尔维茨判据可知，系统稳定的充分必要条件是：

$$\Delta_1=a_1>0$$

$$\Delta_2=\begin{vmatrix} a_1 & a_0 \\ a_3 & a_2 \end{vmatrix}=a_1a_2-a_0a_3>0$$

$$\Delta_3=\Delta=a_3\Delta_2>0$$

或写为：

$$a_0>0 \quad a_1>0 \quad a_2>0 \quad a_3>0$$

$$a_1a_2-a_0a_3>0$$

5.3.3　稳态误差分析

5.3.3.1　稳态误差定义

考虑如图 5.32 所示的控制系统。

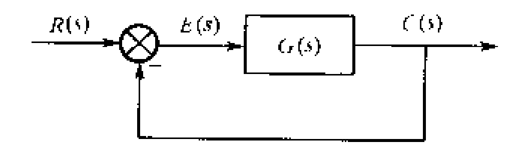


图 5.32　控制系统结构示意图

其闭环传递函数为：

$$\frac{C(s)}{R(s)}=\frac{G(s)}{1+G(s)} \tag{5-74}$$

系统的误差$e(t)$一般定义为被控量的期望值与实际值之差，即

$$e(t)=r(t)-c(t) \tag{5-75}$$

则误差$E(s)$与输入信号$R(s)$之间的传递函数为：

$$\frac{E(s)}{R(s)}=1-\frac{C(s)}{R(s)}=\frac{1}{1+G(s)} \tag{5-76}$$

则$E(s)$为：

$$E(s)=\frac{1}{1+G(s)}R(s) \tag{5-77}$$

误差响应$e(t)$与系统输出响应$c(t)$一样，也包含暂态分量和稳态分量两部分，对于一个稳定系统，暂态分量随着时间的推移将逐渐消失，需要关注的是控制系统平稳以后的误差，即系统误差响应的稳态分量（稳态误差），记为e_{ss}。

定义稳态误差为稳定系统误差响应$e(t)$的终值。当时间t趋于无穷时，$e(t)$的极限存在，则稳态误差为：

$$e_{ss}=\lim_{t\to\infty}e(t)=\lim_{s\to 0}sE(s)=\lim_{s\to 0}\frac{sR(s)}{1+G(s)} \tag{5-78}$$

从上式可得两点结论：

- 稳态误差与系统输入信号$R(s)$的具体形式有关；
- 稳态误差与系统的结构和参数有关。

5.3.3.2 稳态误差系数

系统的开环传递函数$G(s)H(s)$可表示为：

$$G(s)H(s)=\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{s^v(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)} \tag{5-79}$$

常按开环传递函数中所含有的积分环节个数来对系统进行分类。把积分环节个数为0, 1, 2…的系统分别称为0型，Ⅰ型，Ⅱ型…系统。

下面看一下几种常见的稳态误差系数，即静态位置误差系数K_p、静态速度误差系数K_v和静态加速度误差系数K_a。

1. 静态位置误差系数K_p

当系统的输入为单位阶跃信号$r(t)=1(t)$时，由式（5-78）有：

$$e_{ss}=\lim_{s\to 0}s\frac{1}{1+G(s)H(s)}\cdot\frac{1}{s}=\frac{1}{1+\lim\limits_{s\to 0}G(s)H(s)}=\frac{1}{1+K_p} \tag{5-80}$$

其中，$K_p=\lim\limits_{s\to 0}G(s)H(s)$，定义为系统静态位置误差系数。

对于0型系统，有：

$$K_p=\lim_{s\to 0}\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)}=K \tag{5-81}$$

$$e_{ss}=\frac{1}{1+K_p}=\frac{1}{1+K} \tag{5-82}$$

对于Ⅰ型或Ⅰ型以上系统，有：

$$K_p=\lim_{s\to 0}\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{s^v(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)}=\infty \tag{5-83}$$

$$e_{ss}=0 \tag{5-84}$$

2. 静态速度误差系数 K_v

当系统的输入为单位斜坡信号 $r(t)=t\cdot 1(t)$ 时，$R(s)=\frac{1}{s^2}$，由式（5-78）有：

$$e_{ss}=\lim_{s\to 0}\frac{1}{1+G(s)H(s)}\cdot\frac{1}{s^2}=\frac{1}{\lim\limits_{s\to 0}sG(s)H(s)}=\frac{1}{K_v} \tag{5-85}$$

其中，$K_v=\lim\limits_{s\to 0}sG(s)H(s)$，定义为系统静态速度误差系数。

对于0型系统，有：

$$K_v=\lim_{s\to 0}s\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)}=0 \tag{5-86}$$

$$e_{ss}=\frac{1}{K_v}=\infty \tag{5-87}$$

对于Ⅰ型系统，有：

$$K_v=\lim_{s\to 0}s\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{s(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)}=K \tag{5-88}$$

$$e_{ss}=\frac{1}{K_v} \tag{5-89}$$

对于Ⅱ型或Ⅱ型以上系统，有：

$$K_v=\lim_{s\to 0}s\frac{K(\tau_1 s+1)(\tau_2 s+1)\cdots(\tau_m s+1)}{s^v(T_1 s+1)(T_2 s+1)\cdots(T_n s+1)}=\infty \tag{5-90}$$

$$e_{ss}=0 \tag{5-91}$$

3. 静态加速度误差系数 K_a

当系统输入为单位加速度信号 $r(t)=\frac{1}{2}t^2\cdot 1(t)$ 时，$R(s)=\frac{1}{s^3}$，则系统稳态误差为：

$$e_{ss}=\lim_{s\to 0}s\frac{1}{1+G(s)H(s)}R(s)=\lim_{s\to 0}s\frac{1}{1+G(s)H(s)}\cdot\frac{1}{s^3}=\frac{1}{\lim\limits_{s\to 0}s^2G(s)H(s)}=\frac{1}{K_a} \tag{5-92}$$

其中，$K_a=\lim\limits_{s\to 0}s^2G(s)H(s)$，定义为系统静态加速度误差系数。

对于0型系统，有 $K_a=0$，$e_{ss}=\infty$。

对于Ⅰ型系统，有 $K_a=0$，$e_{ss}=\infty$。

对于Ⅱ型系统，有 $K_a=K$。

对于III型或III型以上系统，有 $K_a=\infty$， $e_{ss}=0$。

几种常见系统的稳态误差如表5.1所示。

表5.1 几种常见系统的稳态误差

系统类型	位置误差	速度误差	加速度误差
0型系统	$\frac{1}{K_p}$	∞	∞
I型系统	0	$\frac{1}{K_v}$	∞
II型系统	0	∞	$\frac{1}{K_a}$

【例5-16】 已知如图5.33所示的控制系统。

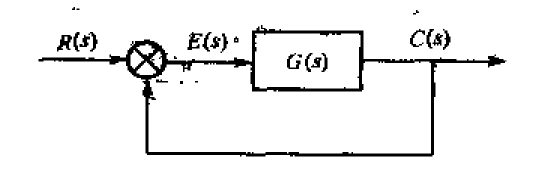


图5.33 控制系统结构示意图

其中， $G(s)=\dfrac{s+5}{s^2(s+10)}$，试计算当输入为单位阶跃信号、单位斜坡信号和单位加速度信号时系统的稳态误差。

解：Simulink的模型如图5.34所示。

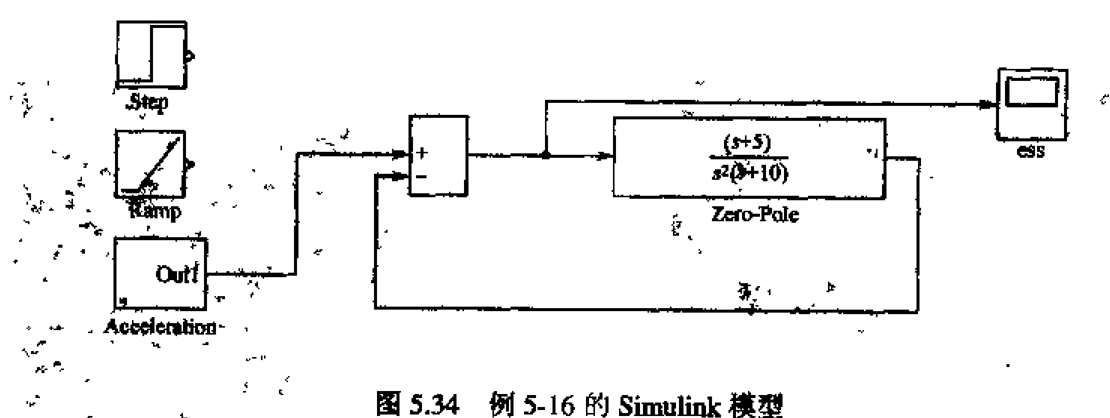


图5.34 例5-16的Simulink模型

图中，“Zero-Pole”建立 $G(s)$ 的模型，信号源选择“Step”(单位阶跃信号)、“Ramp”（单位斜坡信号）和使用基本模块构成的“Acceleration”(单位加速度信号)，“Acceleration”的子系统如图5.35所示，它由单位斜坡信号和 $y=0.5u^2$（u 为输入信号，y 为输出信号）的函数串联而成。

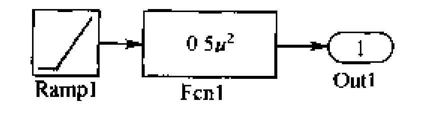


图5.35 例5-16的Acceleration子系统模型

信号源选定“Step”时，连好模型进行仿真，仿真结束后，双击示波器，输出图形如图 5.36 所示，它是输入信号为单位阶跃时系统的输出误差。

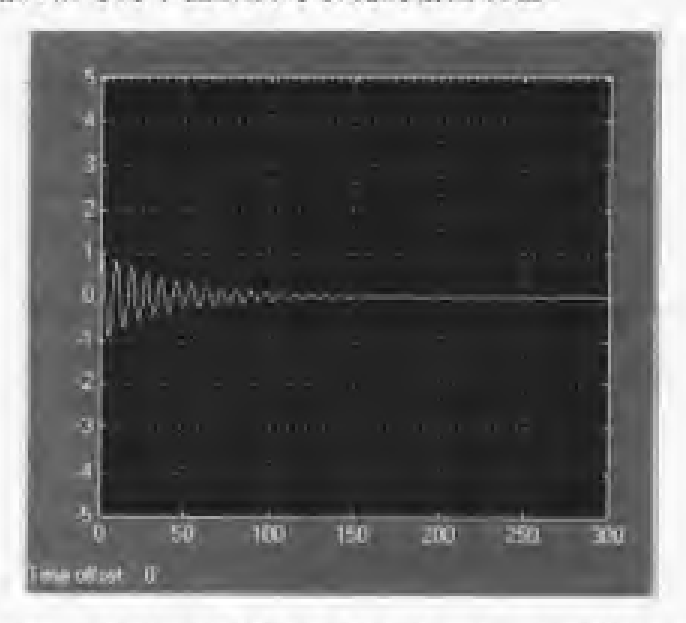


图 5.36　例 5-16 的输入信号为单位阶跃时，系统的输出误差

信号源选定“Ramp”时，连好模型，进行仿真，仿真结束后，双击示波器，输出图形如图 5.37 所示，它是输入信号为单位斜坡时系统的输出误差。

信号源选定“Acceleration”时，连好模型，进行仿真，仿真结束后，双击示波器，输出图形如图 5.38 所示，它是输入信号为单位加速度时系统的输出误差。

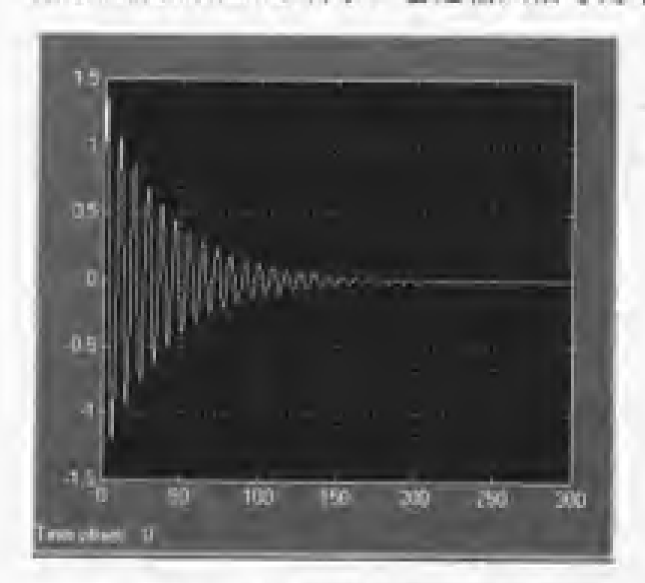


图 5.37　例 5-16 的输入信号为单位斜坡时，系统的输出误差

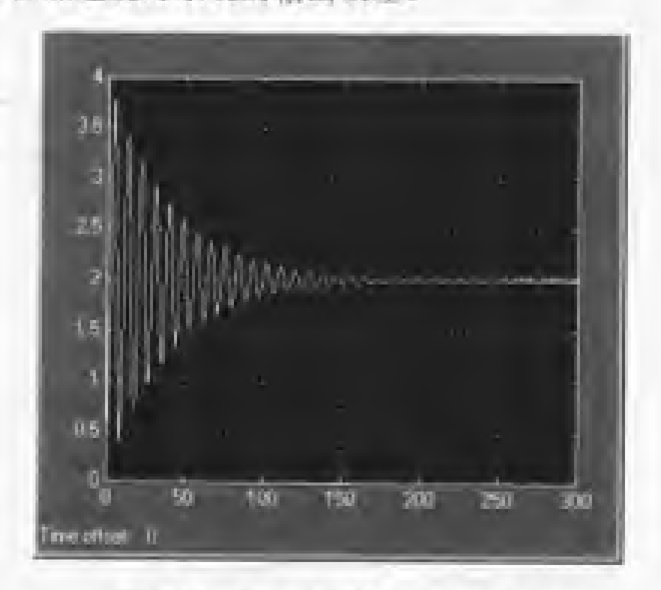


图 5.38　例 5-16 的输入信号为单位加速度时，系统的输出误差

从图 5.36、图 5.37、图 5.38 可以看出不同输入情况下系统的稳态误差，系统是 II 型系统，因此在阶跃、斜坡输入信号下，系统的稳态误差都为零，在加速度信号输入下，存在稳态误差，稳态误差的数值通过放大示波器可以准确地看到，数值为 2，而且仿真时

间越长，离 2 就越接近，这与通过稳态误差系数计算的结果吻合。

5.3.4　MATLAB 在稳定性分析中的应用

MATLAB 提供了直接求取系统所有零极点的函数，因此可以直接根据零极点的分布情况对系统的稳定性进行判断。

MATLAB 提供了直接求根的命令 Roots，因此可以用直接求根来判断稳定性，至于稳定范围的求取，则可以用循环语句迭代计算的方法来求取。下面给出应用实例。

【例 5-17】　已知单位负反馈控制系统的开环传递函数为 $G_o(s)=\dfrac{0.2(s+2.5)}{s(s+0.5)(s+0.7)(s+3)}$，试用 MATLAB 编写程序判断此闭环系统的稳定性，并绘制闭环系统的零极点图。

解：MATLAB 程序代码如下。

```
z=-2.5
p=[0, -0.5, -0.7, -3]
k=0.2
Go=zpk(z, p, k)
Gc=feedback(Go, 1)
Gctf=tf(Gc)
dc=Gctf.den
dens=poly2str(dc{1}, 's')
```

运行结果如下：

```
dens =
 s^4 + 4.2 s^3 + 3 95 s^2 + 1.25 s + 0 5
```

Dens 是系统的特征多项式，接着输入如下 MATLAB 程序代码：

```
den =[1, 4.2, 3.95, 1.25, 0.5]
p=roots(den)
```

运行结果如下：

```
p =
  -3.0058
  -1.0000
  -0.0971 + 0.3961i
  -0.0971 - 0.3961i
```

可见，系统只有负实部的特征根，因此闭环系统是稳定的。

下面绘制系统的零极点图。

MATLAB 程序代码如下：

```
z=-2.5
p=[0, -0.5, -0.7, -3]
k=0 2
```

```
Go=zpk(z, p, k)
Gc=feedback(Go, 1)
Gctf=tf(Gc)
[z, p, k]=zpkdata(Gctf, 'v')
pzmap(Gctf)
grid
```

运行结果如下：

```
z =
   -2.5000
p =
  -3.0058
  -1.0000
  -0.0971 + 0.3961i
  -0.0971 - 0.3961i
k =
      0.2000
```

程序输出如图 5.39 所示。

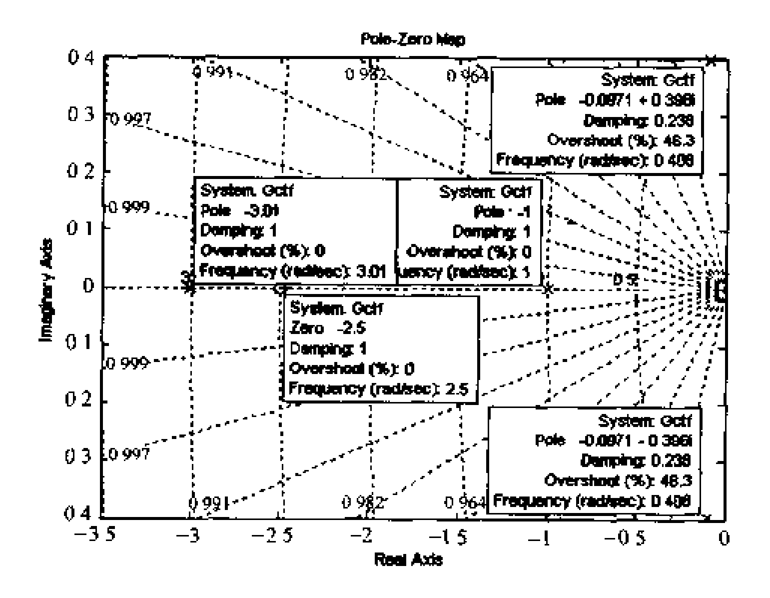


图 5.39　例 5-17 闭环零极点图

第 6 章　根轨迹分析法

6.1　引　　言

本章主要介绍根轨迹法的基本概念以及根轨迹图的基本绘制规则，描述用 MATLAB 绘制根轨迹图的基本方法。通过本章，读者能了解和掌握根轨迹法的概念和绘制方法，熟练使用 MATLAB 绘制根轨迹以及利用根轨迹图对控制系统进行分析。

6.2　根轨迹定义

由前面章节可知，自动控制系统的稳定性完全由它的闭环极点（特征根）决定，而系统的品质则取决于它的闭环极点和零点。因此在设计一个闭环控制系统时，如果能够通过分析开环系统来确定闭环系统的特征，那将具有很大意义。如果系统具有可变的环路增益，则闭环极点的位置取决于所选择的环路增益值，因此，当环路增益变化时，知道闭环极点在 S 平面内如何移动，即根移动的轨迹，那对系统分析和设计具有很大意义。

从系统设计的角度来看，在某些系统中，简单的回路增益调整就可以将闭环极点移动到所需的位置，那么设计问题就转变成了选择合适的增益值的问题。

控制系统的闭环极点就是它的特征方程的根，求解高阶（三阶以上）特征方程的根是很麻烦的，需要借助计算机（MATLAB 可以使该问题变得简单）。但是，求出特征方程的根可能是有限的值，因为当开环增益变化时，特征方程也在变化，因此这种计算是重复进行的。

1948 年，W.R.Evans（伊凡思）根据反馈系统开环和闭环传递函数之间的关系，提出了一种简便的方法，由开环传递函数来直接寻求闭环特征根的轨迹的总体规律，而无须求解高阶系统的特征根。这在工程实践中获得了广泛的应用，这就是根轨迹法。根轨迹法用图解的方法来表示特征方程的根与系统的某个参数（通常是回路增益）之间的全部数值关系，该参数的某个特定值所对应的根显然位于上述关系图上。

当改变增益值或增加开环零极点时，可以利用根轨迹法预测其对闭环极点位置的影响。因此，掌握根轨迹的画法将非常有用，包括手工画和计算机辅助画，它们是利用根轨迹法分析和设计系统的基础。

所谓根轨迹是系统的某个特定参数，通常是回路增益 K 从 0 变化到无穷大时，描绘闭环系统特征方程的根在 S 平面的所有可能位置的图形。

根轨迹法的基本概念是使开环传递函数等于-1 的 s 值必须满足系统的特征方程。

通过考察根轨迹图，设计者能够对控制器的结构和参数做出明智的选择，并推知大量受控系统闭环特性的相关信息。

若能掌握根轨迹图的一般作图规则，那么画已知系统的根轨迹将会变成一件容易的工作。利用 MATLAB 产生根轨迹是一件非常简单的事情，若有手工画根轨迹的经验，那么对于理解 MATLAB 产生的根轨迹图，并迅速获得根轨迹的基本概念，都将是非常有益的。

6.3 根轨迹法基础

当今，在计算机上绘制根轨迹已经是很容易的事，尤其是使用 MATLAB 来绘制根轨迹。计算机绘制根轨迹大多采用直接求解特征方程的方法，也就是每改变一次增益 K 就求解一次特征方程。让 K 从零开始等间隔增大，只要 K 的取值足够多、足够密，相应解特征方程的根就在 S 平面上绘出根轨迹。

传统的根轨迹法是不直接求解特征方程的，它创造了一套行之有效的办法——图解加计算的手工绘图法。如今，尽管手工绘制根轨迹的一些烦琐技艺已经没有多大价值，但是它所发掘出来的根轨迹基本规律，无论用哪种方法作图都是适用的。下面介绍根轨迹最基本、最重要的规律：幅值条件和相角条件。

6.3.1 幅值条件和相角条件

假设控制系统如图6-1所示。

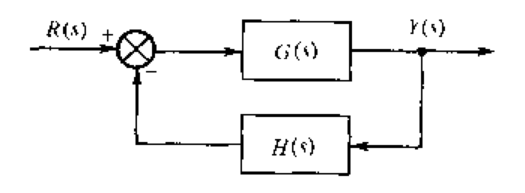


图 6.1 控制系统框图

图中 $G(s)$ 是前向通道的传递函数，表示为：

$$G(s)=K_g\frac{(s+Z_1)(s+Z_2)\cdots(s+Z_i)}{(s+P_1)(s+P_2)\cdots(s+P_j)} \tag{6-1}$$

$H(s)$ 是反馈通道的传递函数，表示为：

$$H(s)=K_h\frac{(s+Z_{i+1})\cdots(s+Z_m)}{(s+P_{j+1})\cdots(s+P_n)} \tag{6-2}$$

反馈通道总假定为负反馈，除非另有说明。

可求得系统闭环传递函数为：

$$\begin{aligned}\frac{Y(s)}{R(s)}&=\frac{G(s)}{1+G(s)H(s)}\\&=\frac{K_g(s+Z_1)(s+Z_2)\cdots(s+Z_i)(s+P_{j+1})\cdots(s+P_n)}{(s+P_1)\cdots(s+P_n)+K_gK_h(s+Z_1)\cdots(s+Z_m)}\end{aligned} \tag{6-3}$$

故闭环零点：$-Z_1\cdots-Z_i$ 和 $-P_{j+1}\cdots-P_n$ 分别是开环传递函数 $G(s)$ 的零点和 $H(s)$ 的极点，

这可从开环传递函数中直接得到。

闭环极点可以从求解下列闭环特征方程中得到：

$$D(s)=(s+P_1)\ldots(s+P_n)+K_g K_h(s+Z_1)\ldots(s+Z_m)=0 \tag{6-4}$$

$D(s)$ 是一个高阶代数方程，甚至没有解析解，求根很不方便，而且直接求根不容易看出闭环极点和系统参数之间的关系，也就是说，从系统参数很难看出它对系统性能的影响。

采用根轨迹法时，把式（6-4）写成另一个等价形式，见式（6-5），称为根轨迹方程。

$$\frac{K\prod_{i=1}^{m}(s+Z_i)}{\prod_{j=1}^{n}(s+P_j)}=-1，\text{其中} K=K_g K_h \tag{6-5}$$

一般意义上的根轨迹，即上述方程在开环增益 K 从 $0\to\infty$ 变化时，闭环极点在复 S 平面内（其中 $S=\sigma+\mathrm{j}\omega$）的变化情况，也就是 180° 根轨迹。

为了便于用作图法求取根轨迹方程的解，把式（6-5）分解成幅值和相角两个方程，分别称为幅值条件和相角条件。

幅值条件：

$$\frac{K\prod_{i=1}^{m}|(s+Z_i)|}{\prod_{j=1}^{n}|(s+P_j)|}=1 \tag{6-6}$$

相角条件：

$$\sum_{i=1}^{m}\angle(s+Z_i)-\sum_{j=1}^{n}\angle(s+P_j)=\pm180^\circ(2q+1),\ q=0,1,2,\ldots \tag{6-7}$$

满足幅值条件和相角条件的所有 s 值，就是特征方程的根，也就是闭环极点。例如，对于一个开环系统，$P_i(i=1,2,3,4)$ 是开环系统的极点，z_1 是开环系统的零点，如图 6.2 所示，对于 S 平面上的试验点 s，如果它在根轨迹上，就应当满足以下相角条件：

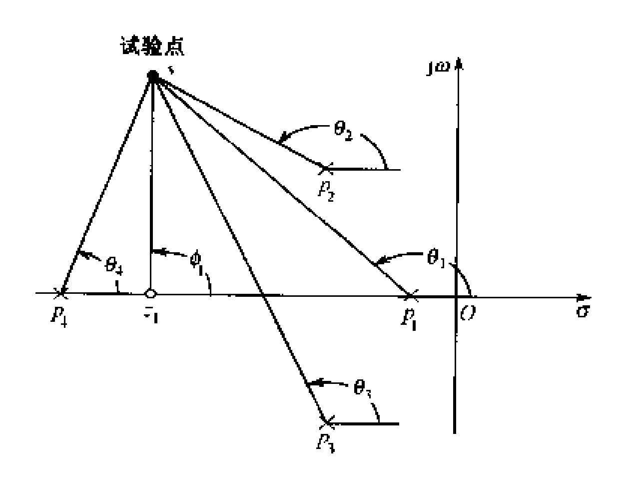


图 6.2 相角条件的图示

$$\phi_1 - \theta_1 - \theta_2 - \theta_3 - \theta_4 = \pm(2k+1)180^\circ \qquad k = 0,1,2,\cdots$$

量出或计算出 5 个角度，就知道试验点 s 是否在根轨迹上。

因为 K 在 $0 \to \infty$ 范围内连续变化，总有一个 K 值能满足幅值条件，因此绘制根轨迹的依据是相角条件，即特征方程的所有根都应满足式（6-7），即相角的和应等于 $\pm180^\circ(2q+1)$。换句话说，在 S 平面内所有满足式（6-7）的 s 点都是系统的特征根，这些点的连线就是根轨迹。

6.3.2 绘制根轨迹的一般法则

显然，用图解分析法来绘制根轨迹图是不方便的，于是发展了一套用于在复平面上快速确定根轨迹基本走向和特殊点（区域）的方法。由此，可以快速地在复平面上画出根轨迹图，这些图在特殊点（区域）上是准确的，其余部分是近似的，可用相角条件来局部准确化。这些方法在系统分析中的另一重要作用是可以从系统的开环零极点分布情况，快速地估计出闭环根轨迹的走向。以下就是这些基本法则。

（1）法则 1：根轨迹的分支数、连续性和对称性。

根轨迹的分支数等于闭环特征方程式的阶次，一般情况下等于开环极点数。根轨迹在复平面上是一簇连续的曲线，并对称于实轴。因为根轨迹是闭环特征方程的根，特征方程的根是实根（在实轴上）或者是共轭复根（对称于实轴），所以根轨迹一定对称于实轴。

（2）法则 2：根轨迹的起点和终点。

根轨迹起始于开环极点，终止于开环零点。如果开环极点数和零点数不等，则其余的根轨迹不是终止于无穷远处，就是起始于无穷远处。

因为根轨迹是闭环特征方特的根，当 $K=0$ 时方程的根就是它的 n 个开环极点，当 $K \to \infty$ 时方程的根就是它的 m 个开环零点。根轨迹的起点和终点是根轨迹的特殊点。当 $n=m$ 时，开始于 n 个开环极点的 n 支根轨迹正好终止于 m 个开环零点。

当 $n>m$ 时，开始于 n 个开环极点的 n 支根轨迹，有 m 支终止于开环零点，有 $n-m$ 支终止于无穷远处。这时，无穷远处也称为"无穷远零点"。

当 $n<m$ 时，终止于 m 个开环零点为 m 支根轨迹，有 n 支来自 n 个开环极点，有 $m-n$ 支来自无穷远处。必须指出，实际系统极少有 $n<m$ 的情况，但是在处理特殊根轨迹时，常常将系统特征方程变形，变形后的等价系统可能会出现这种情况。

（3）法则 3：位于实轴上的根轨迹。

若实轴段右侧开环极点和开环零点数之和为奇数，则实轴段为根轨迹或根轨迹的一部分，如图 6.3 所示。

（4）法则 4：趋于无穷远的根轨迹的渐近线。

趋于无穷远的根轨迹的渐近线均交于实轴上，实轴交点坐标为：

$$\sigma = \frac{\sum_{j=1}^{n}(-P_j) - \sum_{i=1}^{m}(-Z_i)}{n-m} \tag{6-8}$$

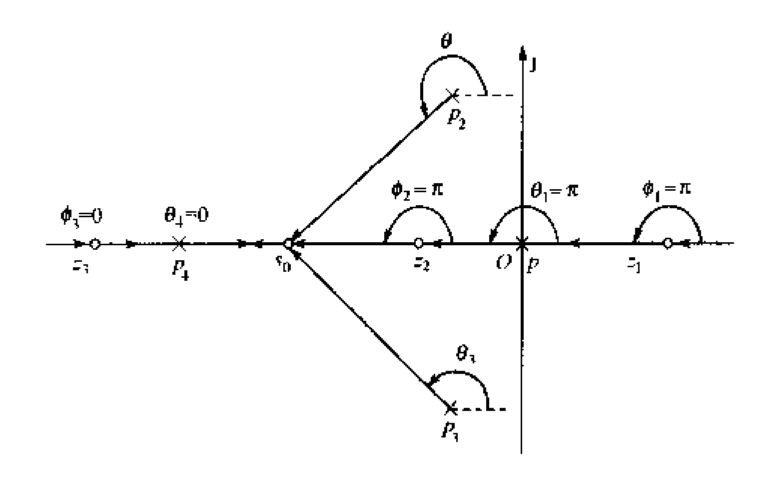


图 6.3 位于实轴上的根轨迹

渐近线与正实轴的夹角为：

$$\varphi = \frac{\pm 180^\circ (2q+1)}{n-m},\quad q = 0,1,2\ldots \tag{6-9}$$

（5）法则 5：根轨迹的分离点和会合点。

当 K 从 0 变到无穷大时，根轨迹可能先会合后分离，这样的点称分离点。分离点对应重闭环极点。

显然，位于实轴上的两个相邻的开环极点之间一定有分离点，因为任何一条根轨迹不可能开始于一个开环极点而终止于另一个开环极点。同理，位于实轴上的两个相邻的开环零点之间也一定有分离点。

当然，分离点也可以是复数，两个相邻的开环复极点（或零点）之间可能有分离点。由根轨迹方程可得分离点和会合点的求解公式如下：

$$k = -\frac{\prod_{j=1}^{n}(s+P_j)}{\prod_{i=1}^{m}(s+Z_i)} \tag{6-10}$$

根轨迹分离点和会合点坐标满足以下方程：

$$\mathrm{d}k/\mathrm{d}s = 0 \tag{6-11}$$

（6）法则 6：根轨迹的起始角和终止角。

根轨迹从开环极点出发的方位角（起始角）为：

$$\varphi = \pm 180^\circ - \sum_{j=1}^{n-1}\varphi_j - \sum_{i=1}^{m-1}\theta_i \tag{6-12}$$

式中，φ_j 是其他开环极点到该极点的方位角，θ_i 是开环零点到该极点的方位角。

根轨迹终止于开环零点的方位角（终止角）上：

$$\theta = \pm 180^\circ + \sum_{j=1}^{n}\varphi_j - \sum_{i=1}^{m-1}\theta_i \tag{6-13}$$

式中，φ_j 是其他开环极点到该零点的方位角，θ_i 是开环零点到该零点的方位角。

（7）法则 7：根轨迹与虚轴交点坐标。

系统闭环特征方程为：

$$D(s)=k\prod_{i=1}^{m}(s+Z_i)+\prod_{j=1}^{n}(s+P_j)=0 \tag{6-14}$$

交点满足下列方程：

$$D(\mathrm{j}\omega)=0 \tag{6-15}$$

（8）法则 8：根轨迹上任一点所对应的根轨迹增益如下。

$$k=\prod_{j=1}^{n}\left|s+P_j\right|/\prod_{i=1}^{m}(s+Z_i) \tag{6-16}$$

6.3.3 与根轨迹分析相关的 MATLAB 函数

在 MATLAB 中，对于如图 6-1 所示的 n 阶单输入单输出系统，采用函数 pzmap( ) 绘制系统零极点，通过输入“rlocus(GH)”可得根轨迹图，它描绘了当开环增益 K 从 $0\to\infty$ 变化时，闭环极点在复 S 平面内的变化情况，即系统 GH 的 180° 根轨迹。MATLAB 会计算出根轨迹的 n 条分支，并以其选定的实轴和虚轴绘制图形。值得注意的是，绘制根轨迹时，应令 S 平面实轴和虚轴的比例尺相同，只有这样才能正确反映 S 平面上坐标位置与相角的关系，在 MATLAB 中，通过“axis equal”命令使实轴和虚轴的比例尺保持相同。在画出根轨迹后，可交互地利用 rlocfind 命令来确定用户鼠标所点之根轨迹上任意点对应的 K 值，K 值所对应的所有闭环极点值也可以利用形如[K, poles] ＝ rlocfind(GH)的命令来显示。

0° 根轨迹对应于图 6-1 中的正反馈或者开环增益 K 为负值的情形。在传递函数前面插入一个负号，使用命令 rlocus(−GH)即可绘制系统 GH 的 0° 根轨迹。

下面介绍与根轨迹分析相关的 MATLAB 函数。

1. 绘制零极点的函数 pzmap( )

调用格式：

```
pzmap(sys)
pzmap(sys1, sys2, …)
[p, z] = pzmap(sys)
```

使用说明：

pzmap( )函数可绘出线性时不变（LTI）系统的零极点图。对单输入单输出（SISO）系统而言，绘制传递函数的零极点；对多输入多输出（MIMO）系统而言，绘制系统的特征矢量和传递零点。

pzmap(sys)函数计算线性时不变（LTI）系统的零极点，并把零极点绘制在复平面上。

pzmap(sys1, sys2, …)函数可在同一个复平面中画出多个 LTI 系统的零极点，为区分各个系统的零极点，可以用不同的颜色来显示，如 pzmap(sys1, 'r', sys2, 'y', sys3, 'g')。

[p, z] ＝ pzmap(sys)返回系统零极点位置的数据，而不直接绘制零极点图，如果需要绘制零极点图可以再用 pzmap(z, p)函数来实现。

【例 6-1】　使用 MATLAB 画出系统的零极点，该系统的闭环传递函数如下：

$$G(s)=\frac{2.5(s+6)}{(s^2+2s+3)(s+5)}$$

解：MATLAB 程序代码如下。

```
num=[2.5, 15];
den=conv([1, 2, 3], [1, 5]);
sys=tf(num, den)
%绘制零极点图
pzmap(sys)
%输出零极点
[p, z]=pzmap(sys)
title('零极点图')
```

程序执行后，输出零极点图，如图 6.4 所示。

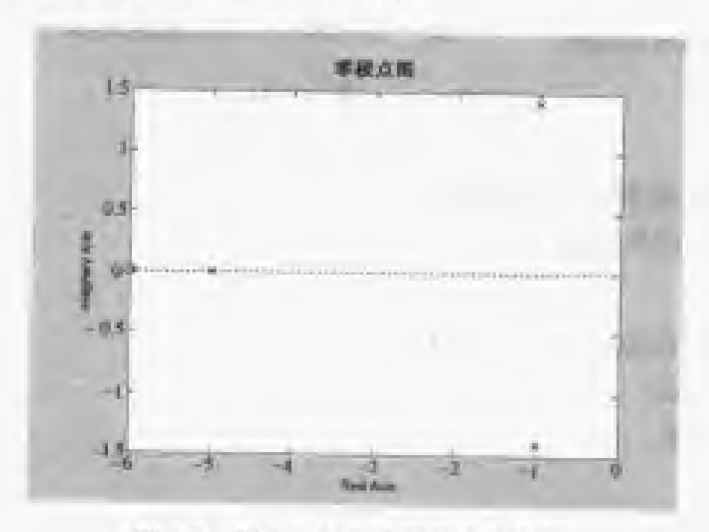


图 6.4　例 6-1 的闭环零极点分布图

程序运行结果如下：

```
p =
   -5.0000
   -1.0000 + 1.4142i
   -1.0000 - 1.4142i
z =
   -6
```

2. 绘制根轨迹的函数 rlocus( )

调用格式：

```
rlocus(sys)
rlocus(sys, k)
```

rlocus(sys1, sys2, …)

[r, k] = rlocus(sys) 或者 r = rlocus(sys, k)

使用说明：

rlocus 计算并绘制 SISO 系统的根轨迹。根轨迹用于研究改变反馈增益对系统极点分布的影响，从而进行系统时域和频域响应的分析。rlocus( )函数既适用于连续时间系统，也适用于离散时间系统。

rlocus(sys, k)绘制增益为 k 时的闭环极点。

rlocus(sys1, sys2, …)在同一个复平面中画出多个 SISO 系统的根轨迹，为区分各个系统的根轨迹，可以用不同的颜色来显示，如 rlocus(sys1, 'r', sys2, 'y:', sys3, 'gx')。

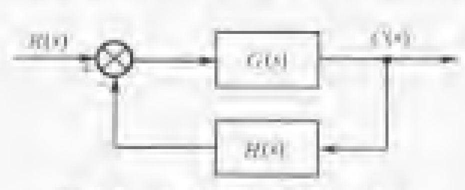


图 6.5　反馈控制系统结构图

[r, k] = rlocus(sys) 或者 r = rlocus(sys, k) 返回增益为 k 时复根位置的矩阵 *R*，*R* 有 length(k) 行，其第 j 行列出的是增益 K(j)时的闭环根。

【例 6-2】　已知如图 6.5 所示的单位负反馈系统，系统的开环传递函数如下：

$$GH(s)=\frac{K(s+1)}{s(0.5s+1)(4s+1)}$$

试使用 MATLAB 绘制系统的根轨迹。

解：MATLAB 程序代码如下：

```
num=[1, 1];
den=conv([1, 0], conv( [0.5, 1], [4, 1]) );
sys=tf(num, den)
%绘制根轨迹图
rlocus(sys)
title('根轨迹图')
```

程序执行后可得如图 6.6 所示的根轨迹。

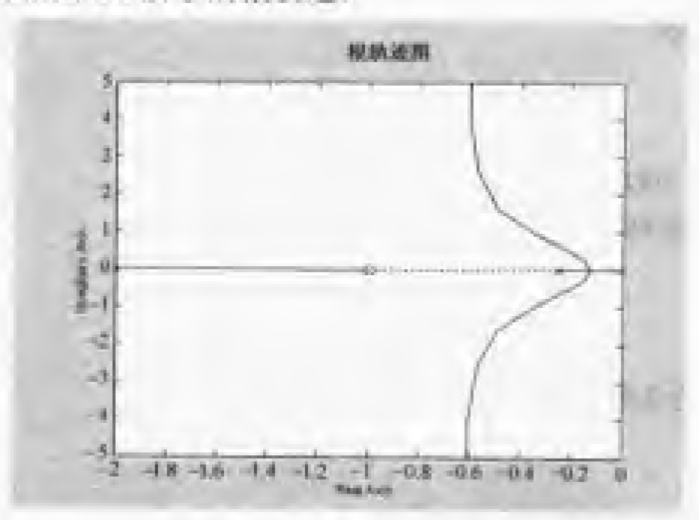


图 6.6　例 6-2 的根轨迹图

3. 计算给定一组根的根轨迹增益的函数 rlocfind( )

调用格式：

```
[k, poles] = rlocfind(sys)
[k, poles] = rlocfind(sys, p)
```

使用说明：

rlocfind( )函数可计算出与根轨迹上极点相对应的根轨迹增益，它适用于连续时间系统和离散时间系统。

[k, poles] = rlocfind(sys)执行后，在根轨迹图形窗口中显示十字形光标，当用户在根轨迹上选择一点时，其相应的增益由 k 记录，与增益相关的所有极点记录于 poles 中。

[k, poles] = rlocfind(sys, p)函数可对指定根计算对应的增益与根矢量 p。

【例 6-3】 已知某单位负反馈系统开环传递函数如下：

$$G(s)=\frac{k(s+5)}{(s+1)(s+3)(s+12)}$$

试绘制系统的根轨迹，并在根轨迹图上任选一点，计算该点的增益 k 及其所有极点的位置。

解：MATLAB 程序代码如下。

```
num=[1, 5];
den=conv([1, 1], conv( [1, 3], [1, 12]) );
sys=tf(num, den)
%绘制根轨迹图
rlocus(sys)
[k, poles]=rlocfind(sys)
%计算用户所选定的点处的增益和其他闭环极点
title('根轨迹图')
```

程序执行后输出如图 6.7 所示的根轨迹图，并在图形窗口中显示十字形光标，当用鼠标左键在根轨迹图上选择一点时，就可得到该点对应的增益 k，以及该 k 值下其他的极点，所有的极点在图中以‘+’表示。

例 6-3 的程序运行结果如下：

```
Select a point in the graphics window
selected_point =
   -4.3294 + 5.0311i
k =
    56.7396
poles =
   -7.2050
   -4.3975 + 5.0034i
   -4.3975 - 5.0034i
```

运行结果如图 6.7 所示。

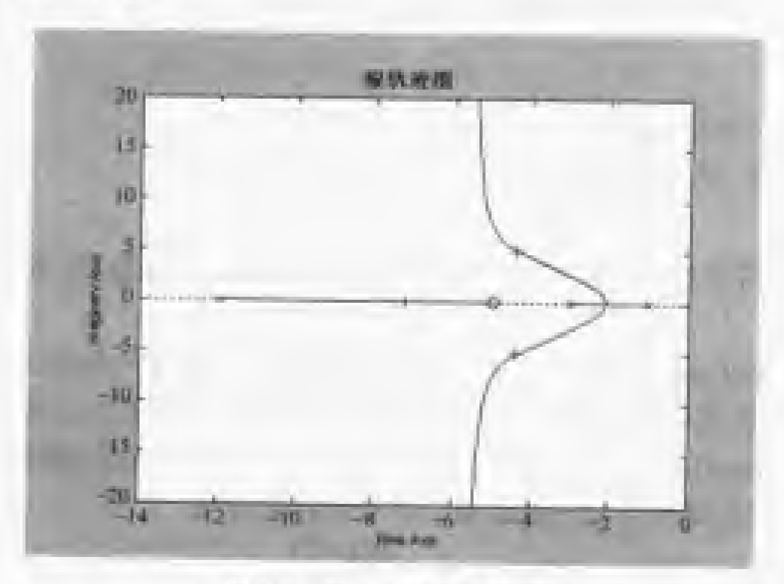


图 6.7　例 6-3 的根轨迹图

由程序执行结果可知，当 $k=56.7396$ 时，该单位负反馈系统的三个闭环极点是：

$$p_1=-7.2050,\ p_2=-4.3975+5.0034\mathrm{i},\ p_3=-4.3975-5.0034\mathrm{i}$$

4. 在连续系统根轨迹图上加等阻尼线和等自然振荡角频率线的函数 sgrid( )

调用格式：

```
sgrid
sgrid(z, wn)
```

使用说明：

sgrid( )函数命令可在连续系统的根轨迹或零极点图上绘制出栅格线，栅格线由等阻尼系数与自然振荡角频率构成。阻尼线的间隔为 0.1，范围从 0 到 1，自然振荡角频率的间隔为 1rad/s，范围从 0 到 10。在绘制栅格线之前，当前窗口必须有连续时间系统的根轨迹或零极点图，或者该函数必须与函数 pzmap( )或 rlocus( )一起使用。

sgrid(z, wn)函数可以指定阻尼系数 z 与自然振荡角频率 wn。

【例 6-4】 使用 MATLAB 画出如图 6.5 所示的单位负反馈系统的带栅格线的根轨迹图，该系统的开环传递函数如下。

$$GH(s)=\frac{2s^2+5s+1}{s^2+2s+3}$$

解：MATLAB 程序代码如下。

```
num=[2, 5, 1];
den=[1, 2, 3];
sys=tf(num, den)
%绘制根轨迹图
rlocus(sys)
```

```
%添加栅格线
sgrid
title ('带栅格线的根轨迹图')
```

程序执行后得到如图 6.8 所示的根轨迹。

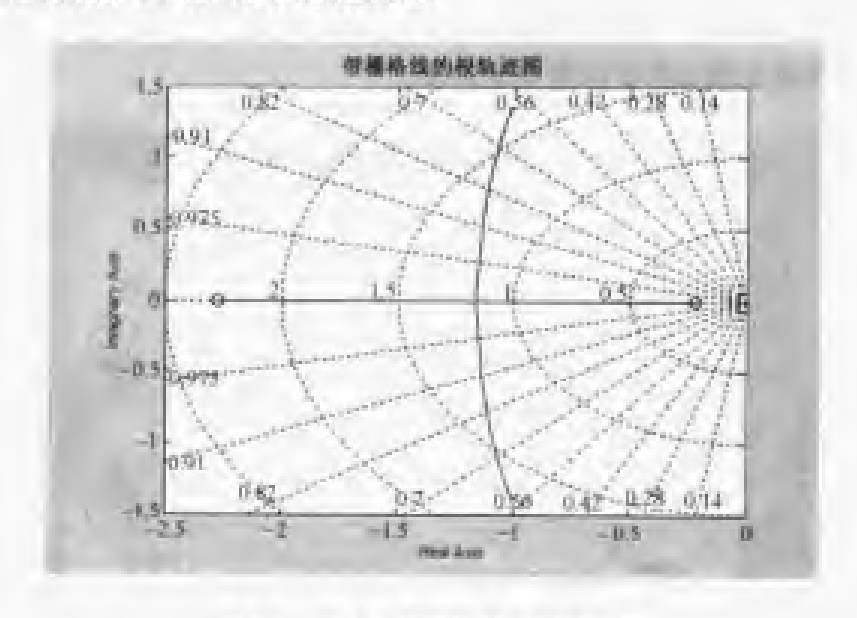


图 6.8 例 6-4 带栅格线的根轨迹图

5．*在离散系统根轨迹图上加等阻尼线和等自然振荡角频率线的函数 zgrid( )*

调用格式：

```
zgrid
zgrid(z, wn)
```

使用说明：

zgrid( )函数可在离散系统的根轨迹或零极点图上绘制出栅格线，栅格线由等阻尼系数与自然振荡角频率构成。阻尼线的间隔为 0.1，范围从 0 到 1，自然振荡角频率的间隔为 p_i/10，范围从 0 到 p_i。在绘制栅格线之前，当前窗口必须有离散时间系统的根轨迹或零极点图。zgrid(z, wn)函数可以指定阻尼系数 z 和自然振荡角频率 wn。

【例 6-5】 使用 MATLAB 画出下面离散系统的带栅格线的根轨迹图，该系统的传递函数如下：

$$H(z)=\frac{2z^2-3.4z+1.5}{z^2-1.6z+0.8}$$

解：MATLAB 程序代码如下。

```
num=[2, -3.4, 1.5];
den=[1, -1.6, 0.8];
%对两个轴用统一的比例尺度
axis('equal')
```

```
%添加栅格线
zgrid('new')
%绘制根轨迹图
rlocus(num, den)
title('带栅格线的离散系统的根轨迹图')
```

程序执行后得到如图 6.9 所示的根轨迹图。

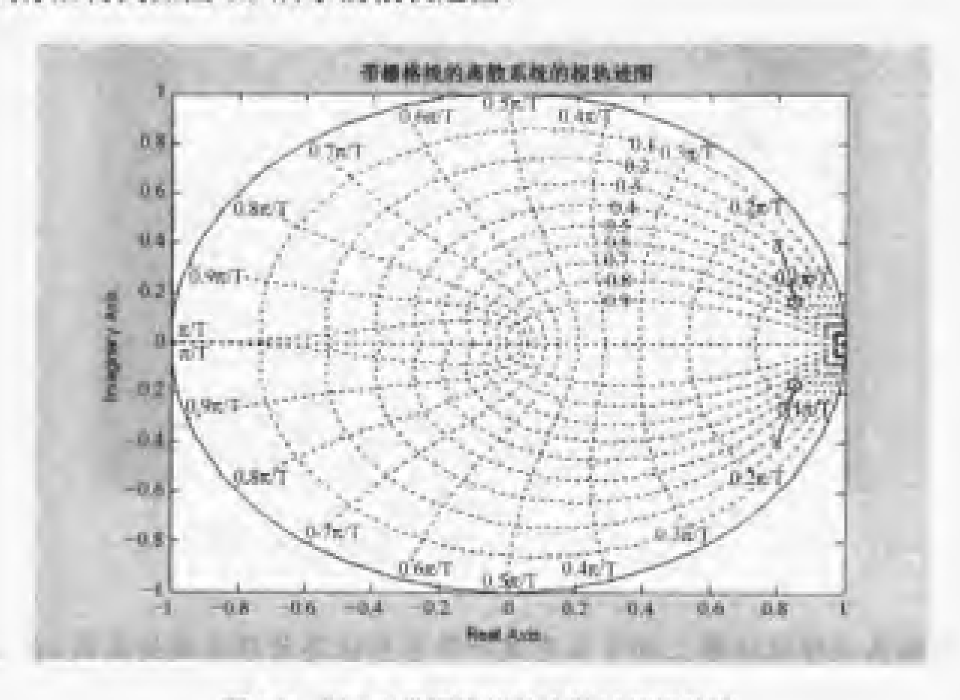


图 6.9　例 6-5 带栅格线的离散系统根轨迹

6.3.4　利用 MATLAB 绘制根轨迹图举例

【例 6-6】　使用 MATLAB 画出如图 6.10 所示的反馈系统的根轨迹，该系统的开环传递函数如下：

$$GH(s)=\frac{K(s+8)}{s(s+2)(s^2+8s+32)}$$

试分析不同的根轨迹特性的适用范围，结合绘制根轨迹的规则，计算 $GH(s)$ 离开上部复极点的角度，并与 MATLAB 绘制的图形进行比较。

求下面两种情况下的 K 值：

（1）两条分支进入右半平面时；

（2）两条分支从复数极点出发在实轴相交时。

解：MATLAB 程序代码如下。

```
num=[1, 8];
den=conv([1, 2, 0], [1, 8, 32]);
sys=tf(num, den)
%计算根轨迹图
rlocus(num, den)
```

```
%调整绘制区域
axis([-15 5 -10 10])
%计算增益值和极点
[k, poles]=rlocfind(sys)
title('根轨迹图')
```

MATLAB 绘制的根轨迹如图 6.10 所示.

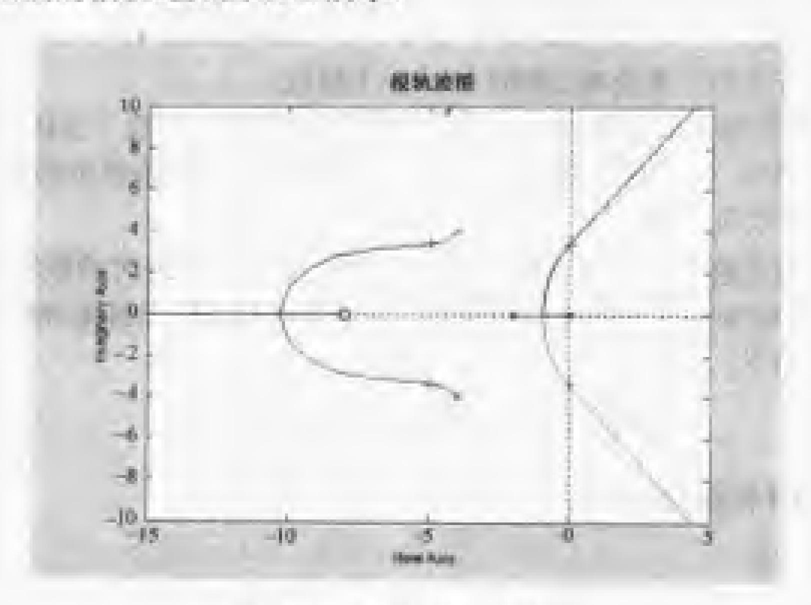


图 6.10　例 6-6 的根轨迹图

可以看到，实轴部分是 $\sigma \leqslant 8$ 和 $-2 \leqslant \sigma \leqslant 0$（对应法则 3），根轨迹有 4 条分支（对应法则 1），分别从开环极点 $s=0, -2, -4+j4$ 和 $-4-j4$ 发出，其中 1 条分支终止于开环零点 $s=-8$ 处，另外 3 条分支趋近 $\pm 60^\circ, -180^\circ$ 的 3 条渐进线（对应法则 2、法则 4），它们的渐进线在点 $\sigma_0=[(0-2-4-4)-(-8)]/(4-1)=-2/3$ 处相交（对应法则 5）。

出射角（对应法则 6）可由以上复数极点算出。首先，可任意指定 $p_1=-4+j4, p_2=-4-j4, p_3=0, p_4=-2, z_1=-8$，然后利用式 $\varphi_1=\arg(p_1-z_1)-[\arg(p_1-p_2)+\arg(p_1-p_3)+\arg(p_1-p_4)]+q\cdot 180^\circ$，代入零点和极点值，可求得出射角为 -116.6° 。

还可以做进一步分析，由于开环极点的数目至少比开环零点的数目多两个，当 K 变化时，闭环极点之和始终为一常数。对于 $K=0$ 时，其和为 $0-2-4-4=-10$，因此对所有的 K 值，4 个闭环极点的和为 -10。因此当两个复数分支穿过虚轴时，其他两个闭环极点的实部必定等于 -5。从根轨迹图可以看出这两个闭环极点为 $s=-5\pm j3.5$ 。

利用 rlocfind 命令，可求得当 $K\approx 45$ 时，从实数极点上出发的两条分支穿入右半平面；当 $K\approx 2070$ 时，从复数极点出发的两条分支到达实轴。

6.4　其他形式的根轨迹

前面研究的是以根轨迹放大系数 K（回路增益）为变量的根轨迹，本节将讨论几种其他形式的根轨迹，包括正反馈系统的根轨迹、参数根轨迹和时滞系统的根轨迹。

6.4.1　正反馈系统的根轨迹

在正反馈条件下，系统的根轨迹方程改为：

$$G(s)H(s)=1 \tag{6-17}$$

幅值条件不变，而相角条件则改为：

$$\sum_{i=-1}^{m}\angle(s+Z_i)-\sum_{j=1}^{n}\angle(s+P_j)=\pm360^\circ\cdot q(\,q=0,1,2...) \tag{6-18}$$

于是与相角有关的三条绘制法则应加以如下修改。

法则 3：关于实轴上的根轨迹，原法则中的奇数应改为偶数（包括 0）。

法则 4 和法则 6：关于根轨迹的渐近线与正实轴的夹角和根轨迹的起始角，原法则中的 $\pm180^\circ(2q+1)$ 应改为 $\pm360^\circ\cdot q$。

该方法也可用来绘制负反馈系统中当开环根轨迹增益 $K<0$ 时的根轨迹。

【例 6-7】　绘制正反馈系统的根轨迹。使用 MATLAB 画出系统的根轨迹，该系统的闭环传递函数如下：

$$GH(s)=\frac{K(s+2)}{(s+3)(s^2+2s+2)},\ K<0\text{。}$$

解：MATLAB 程序代码如下。

```
num=[1, 2];
den=conv([0, 1, 3], [1, 2, 2]);
sys=tf(num, den)
%计算根轨迹图
rlocus(-sys)
%调整绘制区域
axis([-15 5 -10 10])
title('根轨迹图')
```

根轨迹图如图 6.11 所示。

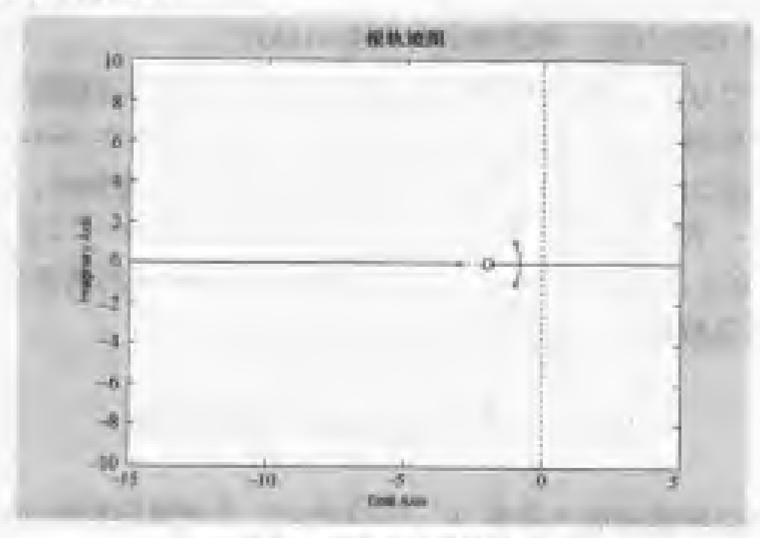


图 6.11　例 6-7 的根轨迹图

6.4.2　参数根轨迹

如前所述，一般的根轨迹的绘制总是设定其变化参数是开环增益 K，当需要研究系统中任一个其他参数变化对系统动力学性能的影响时，就会涉及到参数根轨迹的问题。

设闭环特征方程为：

$$D(\lambda, s)=0 \tag{6-19}$$

式中，λ 就是可变参数。

可将上述方程改写成：

$$\frac{\lambda\prod_{i=1}^{m}(s+Z_i)}{\prod_{j=1}^{n}(s+P_j)}=-1 \tag{6-20}$$

或者

$$G(s)H(s)=\frac{\lambda\prod_{i=1}^{m}(s+Z_i)}{\prod_{j=1}^{n}(s+P_j)} \tag{6-21}$$

式中，$G(s)H(s)$ 称为等效开环传递函数，$-Z_i(i=1,2,\cdots,m)$ 和 $-P_j\ (j=1,2,\cdots,n)$ 分别称为等效的开环零点和开环极点。

注意：它们与实际系统的开环传递函数和开环零极点是不同的。由上述等效根轨迹方程来绘制根轨迹结果就完全一样了。

【例 6-8】　绘制参数 a 的根轨迹，使用 MATLAB 画出系统的根轨迹，该系统的开环传递函数如下：

$$GH(s)=\frac{5(s+a)}{(s+1)(s+3)(s+12)}，其中\ 2\leqslant a\leqslant 10$$

解：MATLAB 程序代码如下。

```
den=conv(conv([1 1], [1 3]), [1 12]);
k=5
%定义数组存储结果
clpoles=[];
param=[];
%a 从 2 变化到 10
for alpha=2:10
num=[0 0 k k*alpha];
clpoly=num+den;
%计算闭环极点
clp=roots(clpoly);
clpoles=[clpoles; clp'];
```

```
param=[param;alpha];
End
%打印 a 和极点表格
disp([param, clpoles])
plot(clpoles, '*')
axis equal;
%调整绘制区域
axis([-4 0 -2 2])
```

根轨迹图如图 6.12 所示。

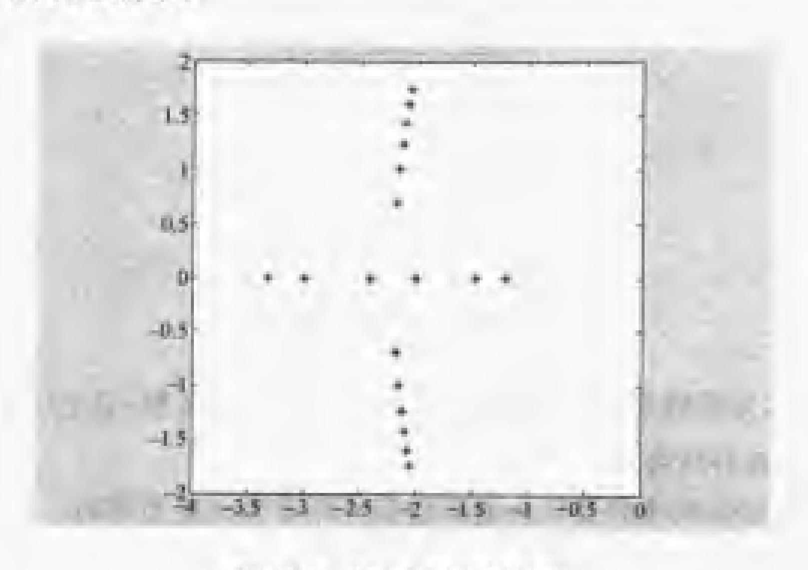


图 6.12　例 6-8 的根轨迹图

6.4.3　时滞系统的根轨迹

如果系统开环传递函数中含有时滞环节，则根轨迹方程可写成：

$$\frac{k\prod_{i=1}^{m}(s+Z_i)}{\prod_{j=1}^{n}(s+P_j)}e^{-\tau s}=-1 \tag{6-22}$$

由上述方程绘制根轨迹是比较麻烦的，使用中常采用一些近似的方法来处理。

对 $e^{-\tau s}$ 进行台劳展开，忽略高次项，可得 $e^{-\tau s}=1-\tau s$；对 $e^{\tau s}$ 进行台劳展开，忽略高次项，可得：

$$e^{-\tau s}\approx\frac{1}{\tau s+1} \tag{6-23}$$

更精确一些的可用所谓的 Pade 展开式：

$$e^{-\tau s}=\frac{1-\frac{\tau s}{2}+\frac{(\tau s)^2}{8}-(\tau s)^3\Big/48\cdots+(-1)^n\frac{(\tau s)^n}{n!2^n}+\cdots}{1+\frac{\tau s}{2}+\frac{(\tau s)^2}{8}+(\tau s)^3\Big/48+\cdots+\frac{(\tau s)^n}{n!2^n}+\cdots} \tag{6-24}$$

如果取一次近似，则

$$e^{-\tau s}=\frac{1-\tau s/2}{1+\tau s/2}=\frac{2-\tau s}{2+\tau s} \tag{6-25}$$

【例 6-9】　绘制时滞系统的根轨迹。使用 MATLAB 画出系统的根轨迹，该系统的开环传递函数如下：

$$GH(s)=\frac{1}{s(s+1)(0.5s+1)}e^{-s}$$

解：MATLAB 程序代码如下。

```
num=[0, 1];
den=conv(conv([1 0], [1 1]), [0.5 1]);
sys1=tf(num, den）
%Pade 近似
[np, dp]=pade(1, 3);
sys=sys1*tf(np, dp）
%绘制根轨迹图
rlocus(sys)
title('时滞系统的根轨迹图')
```

运行程序，输出结果如图 6.13 所示。

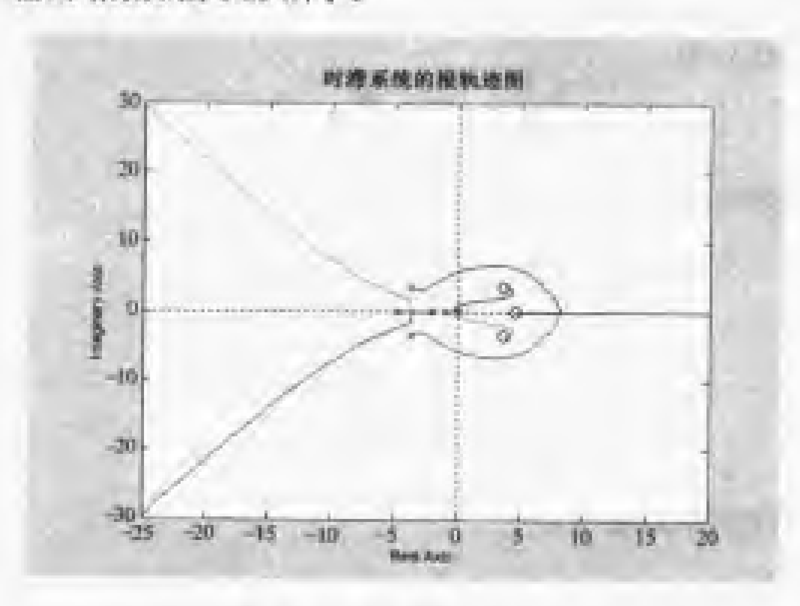


图 6.13　例 6-9 的根轨迹图

6.5　用根轨迹法分析系统的暂态特性

前几节讨论了如何根据开环系统的传递函数求取闭环系统的根轨迹。根轨迹求出后，对于一定的增益 K 值，就可利用幅值条件，确定系统的特征根（闭环极点）。如果闭环系

统的零点、输入信号是已知的，那么可以结合根轨迹，在图上直观地对系统的暂态特性展开分析。用根轨迹法分析系统暂态品质的最大优点是：可以看出在开环放大系数发生变化时，系统的暂态品质是怎样变化的。

从前面章节可得出以下基本结论：

（1）如果已知系统的闭环零极点分布，那么控制系统的动力学性能就可惟一确定。在给出具体输入函数的条件下，可以求出其输出响应和性能指标。

（2）如果所有闭环极点均分布于复平面虚轴左侧，那么系统是稳定的。

（3）稳定系统的动力学特性主要取决于主导极点的位置。所谓主导极点指的是它们距虚轴的距离较其他闭环零极点距虚轴的距离近 5 倍或 5 倍以上。

（4）其他闭环零极点对系统动力学性能的影响：在主导极点的基础上，增加闭环极点，系统的响应速度将降低而超调量将减少；闭环零点的作用则刚好相反，其影响程度将随着距虚轴距离的减小而增强。

（5）偶极子对系统动力学性能的影响可以忽略。所谓偶极子指的是一对靠近的闭环零极点，它们之间的距离较它们本身到虚轴的距离要小 10 倍或 10 倍以上。

由上可知，可以由闭环零极点的分布来分析系统的时域响应。从前面的分析可知，闭环零点是开环传递函数中 $G(s)$的零点和 $H(s)$的极点，可以从开环传递函数中直接得到；而闭环极点，可以在根轨迹图上求出，例如利用 MATLAB 的 rlocfind( )函数可以很方便、直观地求出系统的闭环极点。

【例 6-10】 绘制时滞系统的根轨迹。已知一个三阶系统的结构如图 6.14 所示，系统的开环传递函数 $G(s)$ 为：

$$G(s)=\frac{K}{s(s+1)(s+4)}$$

试从根轨迹图分析其暂态特性。

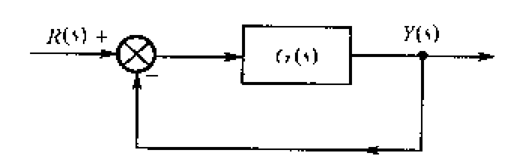


图 6.14 控制系统结构图

解：MATLAB 程序代码如下。

```
num=[0, 4];
den=conv([1, 0], conv( [1, 1], [1, 4]) );
sys=tf(nnm, den)
%绘制根轨迹图
rlocus(sys)
%计算用户所选定的点处的增益和其他闭环极点
[k, poles]=rlocfind(sys)
title('根轨迹图')
```

先画出根轨迹图。利用“rlocfind”命令，可求得当 $K=5$ 时，闭环系统有一对极点位于虚轴上，系统处于稳定边界，输出结果如图 6.15 所示，极点以“+”表示。

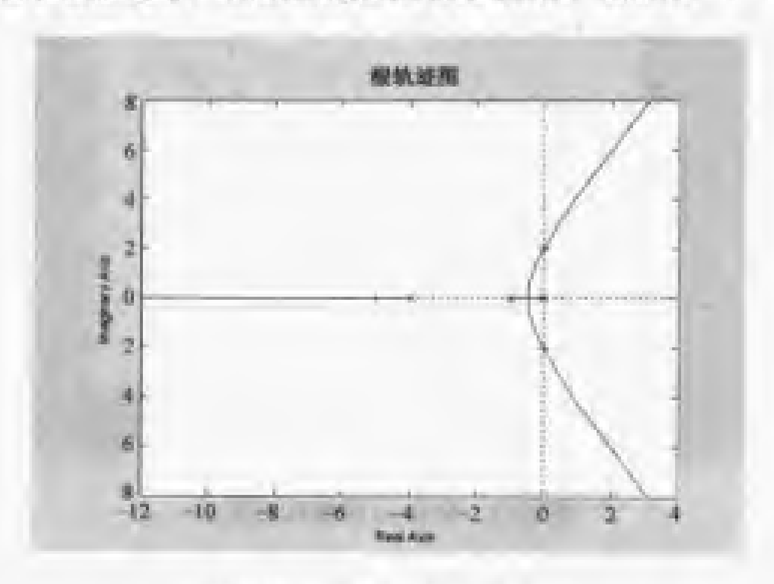


图 6.15　例 6-10 的根轨迹图 1

还可求得当 $K=0.22$ 时，闭环系统的两个极点重合在实轴上，输出结果如图6.16所示，极点以“+”表示。

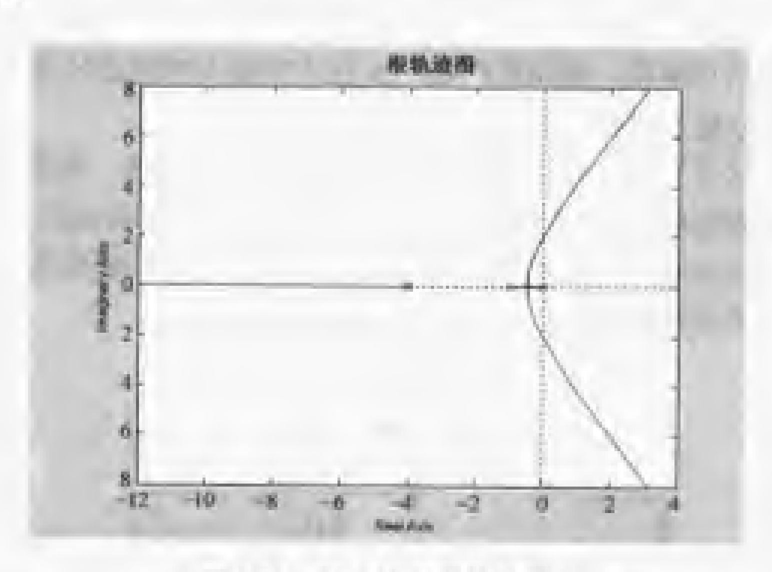


图 6.16　例 6-10 的根轨迹图 2

进一步减小 K 值，会看到其中一个极点将沿实轴向原点靠拢，如图 6-17 所示，暂态响应越来越慢。如果 $K=0.75$，此时一对复极点（$p_1, p_2=-0.39\pm j0.745$）的实部比另一个极点（$p_3=-4.22$）的实部大得多，因此完全可以忽略 p_3 的影响，这样就可以用二阶系统的指标来分析系统的暂态特性。

系统的动态性能最终体现在时间响应上，影响时间响应的因素有两个：闭环传递函数和输入函数。从前面的分析可知：时间响应的暂态分量主要取决于闭环零、极点，时间响应的稳态分量主要取决于输入函数。

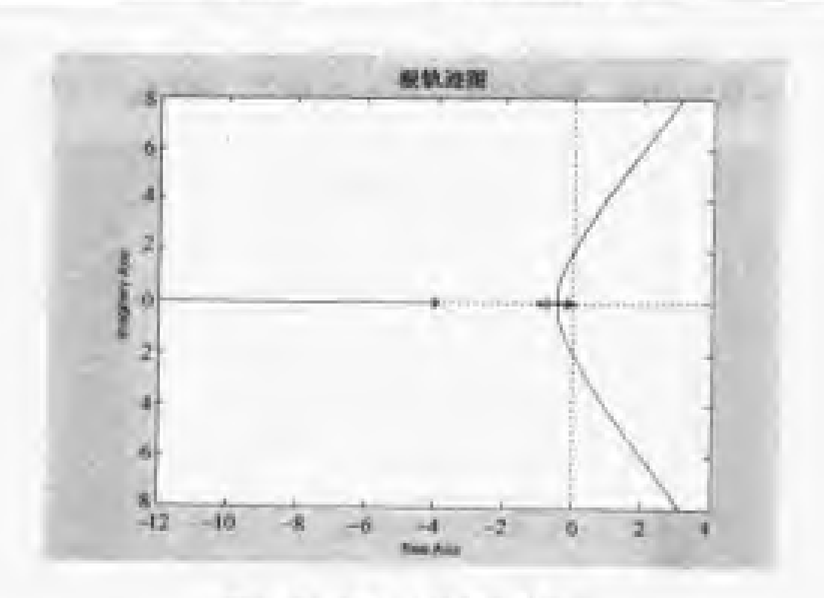


图 6.17　例 6-10 的根轨迹图 3

闭环系统的稳定性完全取决于闭环极点，实际上时间响应的暂态分量也主要取决于闭环极点。每一个闭环极点 p_i 在 S 平面上的位置决定了它对应的暂态分量的运动形式。

图 6.18 表示了 p_i 分布于 S 平面上不同位置所对应的暂态分量，其规律可以总结为：

（1）左右分布决定终值。具体讲就是：p_i 位于虚轴左边时暂态分量最终衰减到零，p_i 位于虚轴右边时暂态分量一定发散，p_i 正好位于虚轴（除原点）时暂态分量为等幅振荡。

（2）虚实分布决定振型。具体讲就是：p_i 位于实轴上时暂态分量为非周期运动，p_i 位于虚轴上时暂态分量为周期运动。

（3）远近分布决定快慢。具体讲就是：p_i 位于虚轴左边时，离虚轴越远过渡过程衰减得越快。所以离虚轴最近的闭环极点“主宰”系统响应的时间最长，是主导极点。

系统的闭环零点对系统的稳定性没有影响，对系统的时间响应没有实质影响，但对时间响应的具体形状是有影响的。

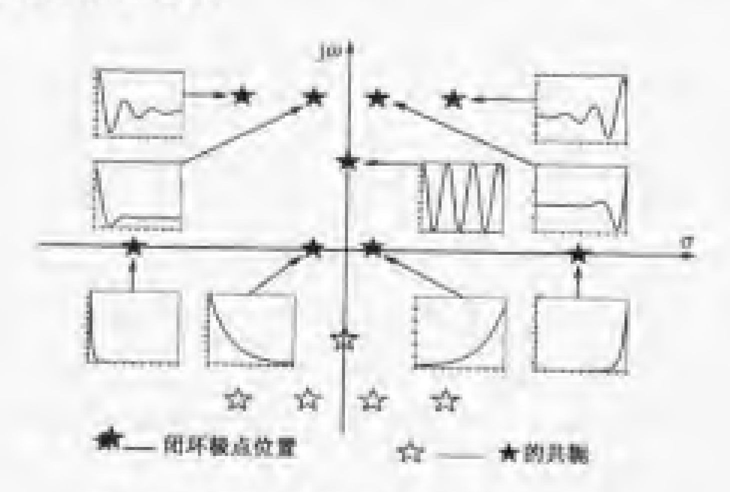


图 6.18　闭环极点分布与暂态分量的运动形式

第 7 章　频域分析法

7.1 引　　言

控制系统的频率特性反映的是系统对正弦输入信号的响应性能。频域分析法是一种图解分析法，它依据系统频率特性对系统的性能（如稳定性、快速性和准确性）进行分析。

频域分析法的突出优点是可以通过试验直接求得频率特性来分析系统的品质，应用频率特性分析系统可以得出定性和定量的结论，并具有明显的物理含义。

通过本章，读者对频率特性的基本概念能有一个比较全面的认识，并学会运用开环频率特性和相应的 MATLAB 工具对系统性能进行频域分析。

7.2 频 率 特 性

频率特性是控制系统的一个重要特性，通过频率特性可间接地对系统动态性能和稳态性能进行分析。

频率特性是控制系统的一种数学模型，可用三种图形对频率特性进行描述，即幅相频率特性曲线（又称极坐标或 Nyquist 曲线）、对数频率特性曲线（又称 Bode 图）、对数幅相频率特性曲线（又称 Nichols 曲线）。

最小相位系统的幅频和相频特性之间存在惟一的对应关系，利用 Nyquist 稳定判据，可由开环频率特性判别闭环系统的稳定性，用相角裕度和幅值裕度来反映系统的相对稳定性。利用等 M 圆和等 N 圆，可由开环频率特性判别闭环频率特性，进而定性或定量地对系统时域响应进行分析。

频域分析法的特点是可以根据开环频率特性来分析闭环系统的性能，并可方便地分析出系统参数对系统性能的影响，从而提出改善系统性能的方法。

频率响应是指系统对正弦输入信号的稳态响应，从频率响应中可获取带宽、增益、转折频率、闭环稳定性等系统特征。

7.2.1 频率特性基本概念

7.2.1.1 频率特性定义

频率特性是指系统在正弦信号作用下，稳态输出与输入之比相对频率的关系特性。频率特性函数与传递函数有直接的关系，记为：

$$G(\mathrm{j}\omega)=\frac{X_{\mathrm{o}}(\mathrm{j}\omega)}{X_{\mathrm{i}}(\mathrm{j}\omega)}=A(\omega)\mathrm{e}^{\mathrm{j}\varphi(\omega)} \tag{7-1}$$

其中，$A(\omega)=\dfrac{X_{\mathrm{o}}(\omega)}{X_{\mathrm{i}}(\omega)}$ 称为幅频特性，$\varphi(\omega)=\varphi_{\mathrm{o}}(\omega)-\varphi_{\mathrm{i}}(\omega)$ 称为相频特性。

频率特性还可表示为：

$$G(\mathrm{j}\omega)=\frac{X_{\mathrm{o}}(\mathrm{j}\omega)}{X_{\mathrm{i}}(\mathrm{j}\omega)}=p(\omega)+\mathrm{j}\theta(\omega) \tag{7-2}$$

式中，，$p(\omega)$ 为 $G(\mathrm{j}\omega)$ 的实部，称为实频特性；$\theta(\omega)$ 为 $G(\mathrm{j}\omega)$ 的虚部，称为虚频特性。

显然：

$$\begin{cases} p(\omega)=A(\omega)\cos\varphi(\omega) \\ \theta(\omega)=A(\omega)\sin\varphi(\omega) \\ A(\omega)=\sqrt{p^2(\omega)+\theta^2(\omega)} \\ \varphi(\omega)=\mathrm{arctg}\dfrac{\theta(\omega)}{p(\omega)} \end{cases} \tag{7-3}$$

需要注意的是，当输入为非正弦的周期信号时，其输入可利用傅里叶级数展开成正弦波的叠加，则其输出为相应的正弦波的叠加。此时系统频率特性定义为系统输出量的傅氏变换与输入量的傅氏变换之比。

7.2.1.2　频率性能指标

与时域响应中衡量系统性能采用时域性能指标类似，频率特性在数值上和曲线形状上的特点通常可用频域性能指标来衡量，它们在很大程度上能够间接地表明系统动静态特性。系统频率特性曲线如图 7.1 所示。

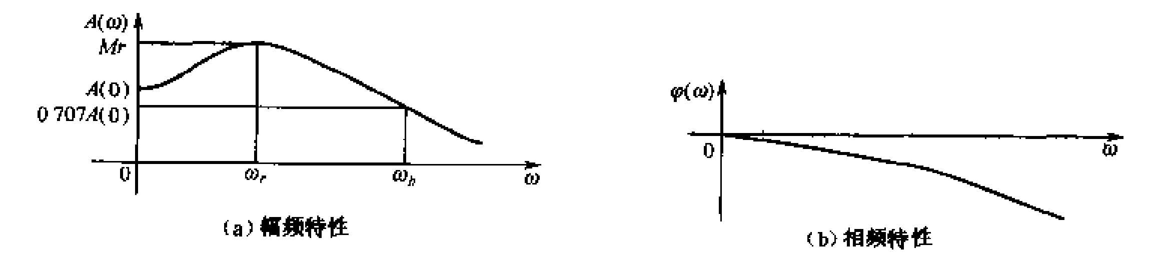


图 7.1　频率特性曲线

1. 谐振频率 ω_r

表示幅频特性 $A(\omega)$出现最大值时所对应的频率。

2. 谐振峰值 M_r

表示幅频特性的最大值，M_r 值大表明系统对频率的正弦信号反映强烈，即系统的平稳性差，阶跃响应的超调量大。

3. 频带 ω_b

表示幅频特性 $A(\omega)$的幅值衰减到起始值的 0.707 倍时所对应的频率。ω_b 大表明系统复现快速变化信号的能力强，失真小，即系统快速性好，阶跃响应上升时间短，调节时间短。

4. 零频 $A(0)$

表示频率$\omega=0$ 时的幅值。$A(0)$表示系统阶跃响应的终值，$A(0)$与 1 之间的差反映了系统的稳态精度，$A(0)$越接近 1，系统的精度越高。

7.2.2 频率响应曲线

频域法作为一种图解分析方法，采用图形化的工具来对系统进行分析。频率特性曲线包括三种常用形式：极坐标图（又称乃奎斯特图或乃氏图或 Nyquist 图）、对数坐标图（又称对数频率特性曲线或 Bode 图）、对数幅相图（又称对数幅相频率特性曲线或 Nichols 图）。

7.2.2.1 极坐标图（Nyquist 图）

系统频率特性可表示为：

$$G(\mathrm{j}\omega)=A(\omega)\mathrm{e}^{\mathrm{j}\varphi(\omega)}$$

用向量表示某一频率 ω_i 下的 $G(\mathrm{j}\omega_i)$ 向量的长度 $A(\omega_i)$，向量极坐标角为 $\varphi(\omega_i)$，$\varphi(\omega_i)$ 的正方向取为逆时针方向，选极坐标与直角坐标重合，极坐标的顶点在坐标原点，如图 7.2 所示。

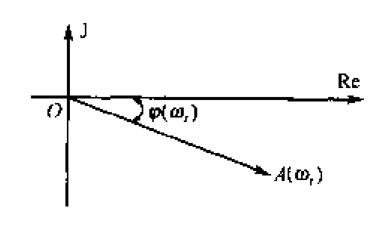


图 7.2 极坐标图

频率特性 $G(\mathrm{j}\omega)$ 是输入频率 ω 的复变函数，是一种变换。当频率 ω 由 $0\to\infty$ 时，$G(\mathrm{j}\omega)$ 变化的曲线，即向量端点轨迹，就称为极坐标图。

极坐标图中，当 $\omega=\omega_i$ 时，在实轴上的投影即为实频特性 $p(\omega_i)$，在虚轴上的投影即为虚频特性。

用频率法分析控制系统时控制系统通常由若干环节组成，根据它们的基本特性，可以把系统分解成一些典型环节的串联，再按照串联的规律将这些典型环节的频率特性组合起来，即得到整个系统的开环频率特性。

绘制系统开环频率特性极坐标图的基本步骤如下所述：

（1）将系统开环传递函数分解成若干典型环节的串联形式；

（2）典型环节幅频特性相乘得到系统开环幅频特性；

（3）典型环节相频特性相加得到系统开环相频特性；

（4）若幅频特性有渐近线，则根据开环频率特性表达式的实部和虚部，求出渐近线；

（5）最后在 $G(\mathrm{j}\omega)H(\mathrm{j}\omega)$ 平面上绘制出系统开环频率特性的极坐标图。

MATLAB 提供了绘制系统极坐标图的函数 nyquist( )，其用法如下。

nyquist(a, b, c, d)：绘制系统的一组 Nyquist 曲线，每条曲线对应于连续状态空间系统[a, b, c, d]的输入/输出组合对，其中频率范围由函数自动选取，且在响应快速变化的位置

会自动采用更多取样点。

nyquist(a, b, c, d, iu)：绘制从系统第 in 个输入到所有输出的极坐标图。

nyquist(num, den)：绘制以连续时间多项式传递函数表示的系统极坐标图。

nyquist(a, b, c, d, iu, w)或 nyquist(num, den, w)：利用指定的角频率矢量绘制系统的极坐标图。

当不带返回参数时，直接在屏幕上绘制出系统的极坐标图（图 7.3 中用箭头表示 w 的变化方向，负无穷到正无穷）。

当带输出变量[re, im, w]引用函数时，可得到系统频率特性函数的实部 re、虚部 im 以及角频率点 w 矢量（为正的部分）；可用 plot(re, im) 绘制出 w 从负无穷到零变化的对应部分。

【例 7-1】　已知一个典型的一阶环节传递函数：

$$G(s)=\frac{5}{3s+1}$$

试绘制该环节的 Nyquist 图。

解：MATLAB 程序代码如下。

```
num=5
den=[3, 1]
G=tf(num, den)
nyquist(G)
grid
```

响应曲线如图 7.3 所示。

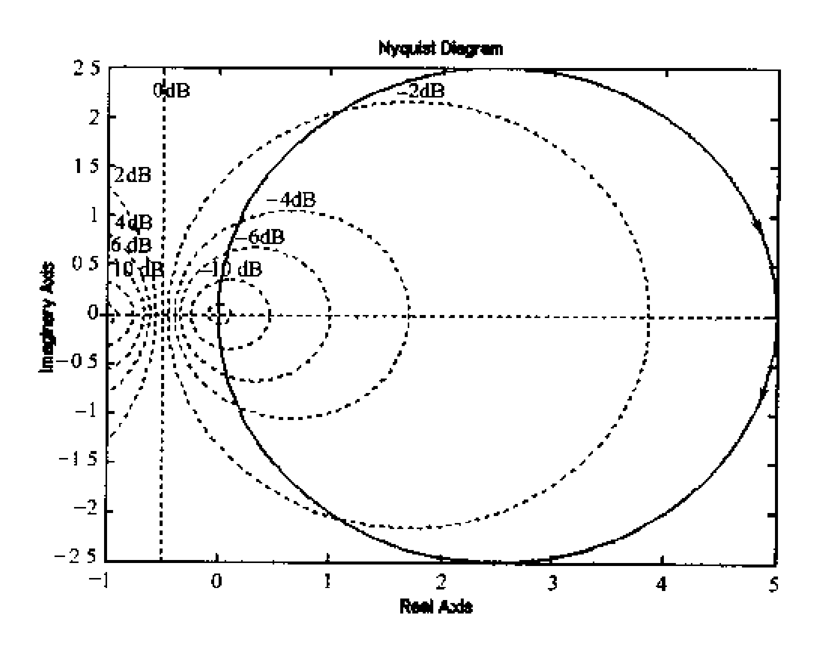


图 7.3　例 5-1 的 Nyquist 曲线

7.2.2.2　对数坐标图（Bode 图）

Bode 图由对应对数幅频特性和对应对数相频特性的两张图组成，如图 7.4 所示。

对数幅频特性是频率特性的对数值 $L(\omega)=20\lg A(\omega)$ 与频率 ω 的关系曲线；对数相频特性是频率特性的相角 $\varphi(\omega)$ 与频率 ω 的关系曲线。

对数幅频特性的纵轴为 $L(\omega)=20\lg A(\omega)$，单位为 dB（分贝），采用线性分度，$A(\omega)$ 每增加 10 倍，$L(\omega)$ 增加 20 dB；值坐标采用对数分度，即横轴上的 ω 取对数后为等分点。

对数相频特性横轴采用对数分度，纵轴为线性分度，单位为°(度)。

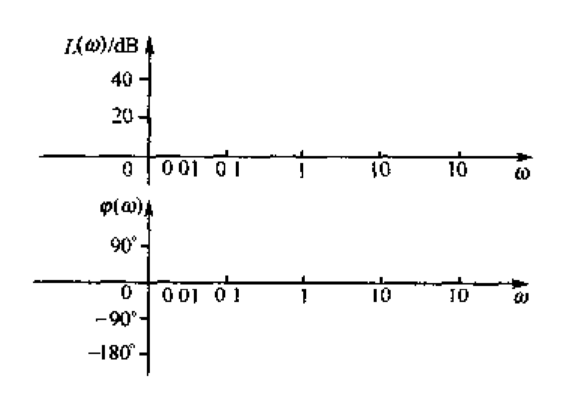


图 7.4　对数坐标图

当 n 个环节串联时：

$$
\begin{aligned}
G(\mathrm{j}\omega)&=G_1(\mathrm{j}\omega)G_2(\mathrm{j}\omega)\cdots G_n(\mathrm{j}\omega)\\
&=A_1(\omega)A_2(\omega)\cdots A_n(\omega)\mathrm{e}^{\mathrm{j}[\varphi_1(\omega)+\varphi_2(\omega)+\cdots+\varphi_n(\omega)]}\\
&=A_1(\omega)\mathrm{e}^{\mathrm{j}\varphi_1(\omega)}A_2(\omega)\mathrm{e}^{\mathrm{j}\varphi_2(\omega)}\cdots A_n(\omega)\mathrm{e}^{\mathrm{j}\varphi_n(\omega)}
\end{aligned}
\tag{7-4}
$$

对数幅频特性 $L(\omega)$ 为：

$$
\begin{aligned}
L(\omega)&=20\lg|G(\mathrm{j}\omega)|\\
&=20\lg A_1(\omega)A_2(\omega)\cdots A_n(\omega)\\
&=20\lg A_1(\omega)+20\lg A_2(\omega)+\cdots+20\lg A_n(\omega)\\
&=L_1(\omega)+L_2(\omega)+\cdots+L_n(\omega)
\end{aligned}
\tag{7-5}
$$

对数相频特性 $\varphi(\omega)$ 为：

$$
\varphi(\omega)=\angle G(\mathrm{j}\omega)=\varphi_1(\omega)+\varphi_2(\omega)+\cdots+\varphi_n(\omega)
\tag{7-6}
$$

可以看出，对数幅频特性采用 $20\lg A(\omega)$，就可以把幅值的乘除运算简化为加减运算，从而简化曲线的绘制过程。

绘制系统开环对数幅频特性的基本步骤如下所述：

（1）将系统的开环传递函数写成典型的环节乘积（即串联）形式；

（2）如果存在转折频率，则在 ω 轴上标出转折频率的坐标位置；

（3）将各串联环节的对数幅频特性叠加，得到系统开环对数幅频特性的渐近线；

（4）修正误差，画出更加精确的对数幅频特性；

（5）画出各串联典型环节的相频特性，将其相加，得到系统开环相频特性。

MATLAB 提供了绘制系统 Bode 图的函数 bode( )，其用法如下。

bode(a, b, c, d)：绘制系统的一组 Bode 图，它们是针对连续状态空间系统[a, b, c, d]的每个输入的 Bode 图，其中频率范围由函数自动选取，且在响应快速变化的位置会自动采用更多取样点。

bode(a, b, c, d, iu)：绘制从系统值 iu 个输入到所有输出的 Bode 图。

bode(num, den)：绘制以连续时间多项式传递函数表示的系统 Bode 图。

bode(a, b, c, d, iu, w)或 bode(num, den, w)：利用指定的角频率矢量绘制系统的 Bode 图。

当带输出变量[mag, pha, w]或[mag, pha]引用函数时，可得到系统 Bode 图相应的幅值

mag、相角 pha 与角频率点 w 矢量，或只是返回幅值与相角。相角以度为单位，幅值可转换为分贝单位：mag(dB)＝20×lg10(mag)。

【例 7-2】　已知一个典型的二阶环节传递函数：

$$G(s)=\frac{\omega_n^{\ 2}}{s^2+2\zeta\omega_n s+\omega_n^{\ 2}}$$

其中 $\omega_n=0.7$，试分别绘制 $\zeta=0.1, 0.4, 1.0, 1.6, 2.0$ 时的 Bode 图。

解：　MATLAB 程序代码如下。

```
w=[0, logspace(-2, 2, 200)]
wn=0.7
tou=[0.1, 0.4, 1.0, 1.6, 2.0]
for j=1:5
sys=tf([wn*wn], [1, 2*tou(j)*wn, wn*wn])
bode(sys, w)
hold on
end
gtext('tou=0.1')
gtext('tou=0.4')
gtext('tou=1.0')
gtext('tou=1.6')
gtext('tou=2.0')
```

响应曲线如图 7.5 所示。

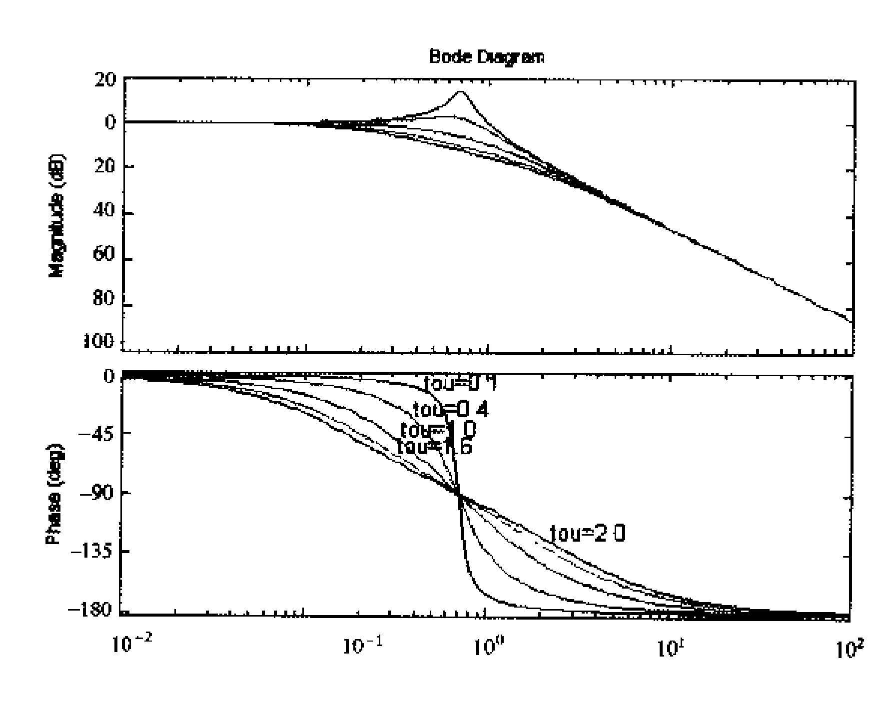


图 7.5　例 7-2 的二阶系统 Bode 图

7.2.2.3　对数幅相图（Nichols 图）

对数幅相图也称 Nichols 图，它是将对数幅频特性和相频特性两张图在角频率为参变量的情况下合成为一张图，如图 7.6 所示。其特点是纵轴为 $L(\omega)=20\lg A(\omega)$，单位为 dB（分贝），采用线性分度；横坐标采用对数分度，单位为°（度），频率 ω 为参变量。

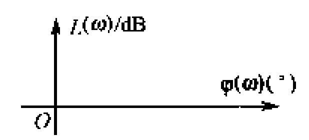


图 7.6　对数幅相坐标系

Nichols 图通常与等 M 圆（等幅值圆）和等 N 圆（等相角圆）一起使用，从开环频率特性获得闭环频率特性。

设开环频率特性 $G(j\omega)$ 为：

$$G(j\omega)=p(\omega)+j\theta(\omega)=x+jy \tag{7-7}$$

则闭环频率特性的幅值为：

$$|M(j\omega)|=\left|\frac{G(j\omega)}{1+G(j\omega)}\right|=\left|\frac{\sqrt{x^2+y^2}}{\sqrt{(x+1)^2+y^2}}\right|$$

令 $M=|M(j\omega)|$，则

$$M\sqrt{(x+1)^2+y^2}=\sqrt{x^2+y^2} \tag{7-8}$$

整理得：

$$(1-M^2)x^2+(1-M^2)y^2-2M^2x=M^2 \tag{7-9}$$

当 M=1 时，由上式可求得 x=−1/2，这是通过点（−1/2，j0）且与虚轴平行的一条直线；

当 M≠1 时，上式可化简为：

$$\left(x-\frac{M^2}{1-M^2}\right)^2+y^2=\left(\frac{M}{1-M^2}\right)^2 \tag{7-10}$$

当给定 M 值（等 M 值）时，上式是一个圆方程式，圆心在 $\left(-\frac{M^2}{M^2-1},\ j0\right)$ 处，半径为 $\left|\frac{M}{M^2-1}\right|$，因此在 $G(j\omega)$ 平面上，等 M 轨迹是一簇圆，如图 7.7 所示。

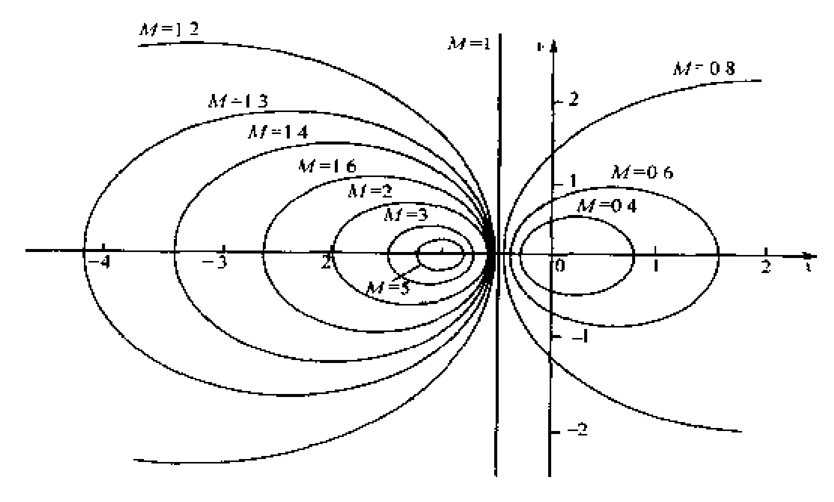


图 7.7　等 M 圆

从图中可以看出：

当 $M>1$ 时，随着 M 值的增大，等 M 圆半径愈来愈小，最后收敛于（−1, j0）点，且这些圆均在 $M=1$ 直线的左侧。

当 $M<1$ 时，随着 M 值的减小，等 M 圆半径也愈来愈小，最后收敛于原点，且这些圆都在 $M=1$ 直线的右侧。

当 $M=1$ 时，它是通过（−1/2, j0）点平行于虚轴的一条直线。等 M 圆既对称于 $M=1$ 的直线，又对称于实轴。

闭环频率特性的相角 φ_m 为：

$$\varphi_m=\angle\frac{C(\mathrm{j}\omega)}{R(\mathrm{j}\omega)}=\mathrm{arctg}\frac{y}{x}-\mathrm{arctg}\frac{y}{x+1} \tag{7-11}$$

令 $N=\mathrm{tg}\varphi_m$，整理得：

$$\left(x+\frac{1}{2}\right)^2+\left(y-\frac{1}{2N}\right)^2=\frac{1}{4}+\left(\frac{1}{2N}\right)^2 \tag{7-12}$$

当给定 N 值（等 N 值）时，上式是一个圆方程式，圆心在 $\left(-\frac{1}{2},\ \frac{1}{2N}\right)$ 处，半径为 $\sqrt{\frac{1}{4}+\left(\frac{1}{2N}\right)^2}$，称为等 N 圆，如图 7.8 所示。

等 N 圆实际上是等相角正切的圆，当相角增加 ±180° 时，其正切相同，因而在同一个圆上。

所有等 N 圆均通过原点和（−1, j0）点，对于等 N 圆，并不是一个完整的圆，而只是一段圆弧。

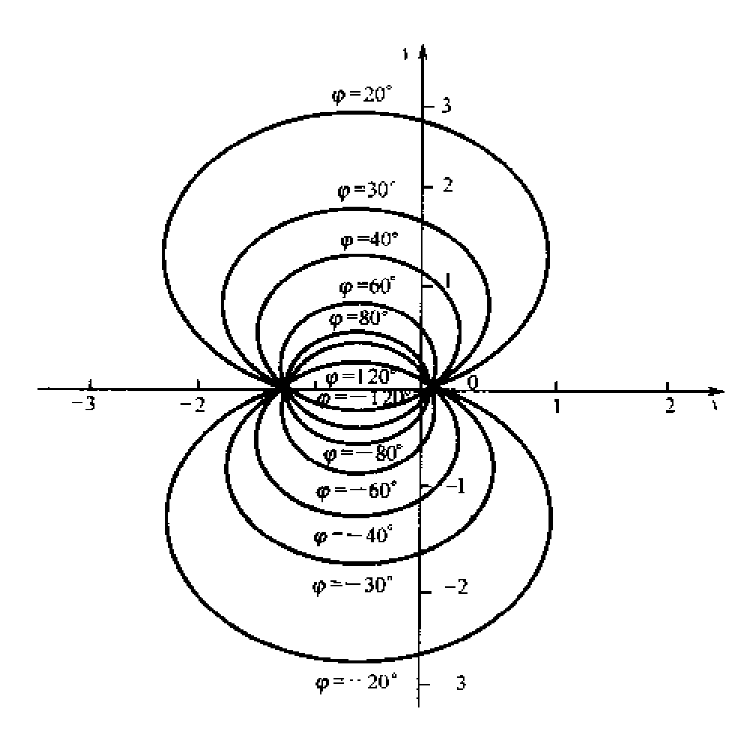


图 7.8　等 N 圆

MATLAB 提供了绘制系统的 Nichols 图的函数 nichols ( )，其用法如下。

nichols (a, b, c, d)：绘制系统的一组 Nichols 图，它们是针对连续状态空间系统[a, b, c, d]的每个输入的 Nichols 图，其中频率范围由函数自动选取，且在响应快速变化的位置会自动采用更多取样点。

nichols (a, b, c, d, iu)：绘制从系统第 iu 个输入到所有输出的 Nichols 图。

nichols (num, den)：绘制以连续时间多项式传递函数表示的系统 Nichols 图。

nichols (a, b, c, d, iu, w)或 nichols (num, den, w)：利用指定的角频率矢量绘制系统的 Nichols 图。

当带输出变量[mag, pha, w]或[mag, pha]引用函数时，可得到系统 Nichols 图相应的幅值 mag、相角 pha 与角频率点 w 矢量，或只是返回幅值与相角。相角以度为单位，幅值可转换为分贝单位：mag(dB)＝20×lg10(mag)。

MATLAB 提供了绘制等 M 圆和等 N 圆的函数 ngrid ( )。ngrid( )对于用 nichols 函数绘制的 Nichols 曲线，绘制的相应的等 M 圆和等 N 圆，均为虚线圆，并提供有关的对数幅值和相位值。值得注意的是，在对数坐标中，圆的形状发生变化。

【例 7-3】 已知一高阶系统的传递函数为：

$$G(s)=\frac{0.0001s^3+0.0281s^2+1.06356s+9.6}{0.0006s^3+0.0286s^2+0.06356s+6}$$

试绘制系统的 Nichols 图。

解：MATLAB 程序代码如下。

```
num=[0.0001, 0.0281, 1.06356, 9.6]
den=[0.0006, 0.0286, 0.06356, 6]
G=tf(num, den)
ngrid('new')
nichols(G)
```

响应曲线如图 7.9 所示。

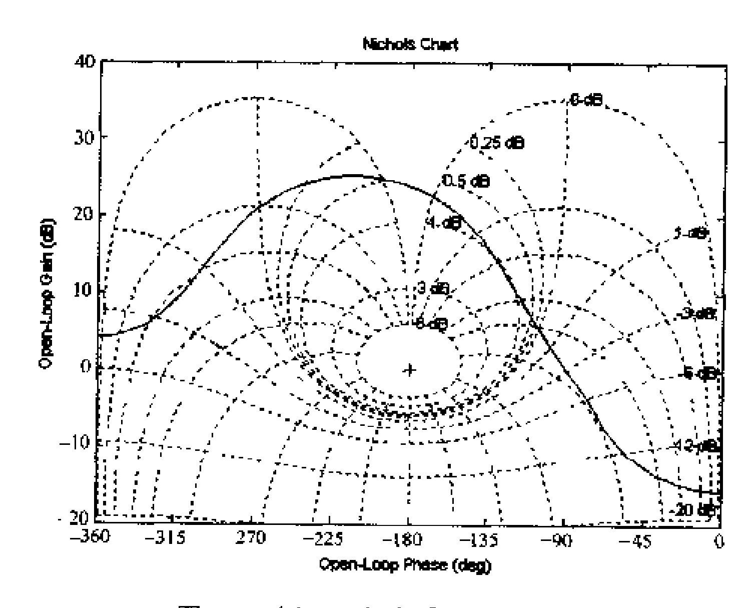


图 7.9　例 7-3 高阶系统 Nichols 图

7.3 频率响应分析

时域分析法是分析控制系统的直接方法，时域分析中的性能指标直观地反映了控制系统的动态响应特征；频域分析法是一种工程上广为采用的、间接的系统分析与综合方法，系统频率特性函数的某些特征可用做间接反映系统动态响应的特征。

7.3.1 系统品质分析

利用频率特性分析系统品质主要包括开环频率特性与时域响应的关系、闭环频域性能指标与时域性能指标的关系。

7.3.1.1 开环频率特性与时域响应的关系

开环频率特性与时域响应的关系通常分为三个频段加以分析，即低频段、中频段和高频段。

1. 低频段

低频段通常指 $L(\omega)=20\lg|G(\mathrm{j}\omega)|$ 的渐近线在第一个转折频率之前的频段，这一频段的特性完全由积分环节和开环放大倍数决定。

低频段对数幅频特性为：

$$L_d(\omega)=20\lg K-20v\lg\omega \quad (7\text{-}13)$$

其中，K 为开环放大倍数，v 为开环传递函数中积分环节的个数。低频段的斜率越小，位置越高，对应系统积分环节的数目越多（系统型号越高），开环放大倍数 K 越大，则在闭环系统稳定的条件下，其稳态误差越小，动态响应的跟踪精度越高。

2. 中频段

中频段指开环对数幅频特性曲线在开环截止频率 ω_c 附近（0dB 附近）的区段，这一频段的特性集中反映了闭环系统动态响应的平稳性和快速性。

时域响应的动态特性主要取决于中频段的形状。

反映中频段形状的三个参数为：开环截止频率 ω_c、中频段斜率、中频段宽度。ω_c 的选择，决定于系统暂态、响应速度的要求；中频段越长，相位裕量越大。

为使系统稳定，且有足够的稳定裕度，一般希望开环对数幅频特性斜率为-20dB/dec，且中频段有足够的宽度；或开环对数幅频特性斜率为-40dB/dec，且中频段较窄。

3. 高频段

高频段指开环对数幅频特性在中频段以后的频段，高频段的形状主要影响时域响应的起始段。

在进行分析时，可以将高频段进行近似处理，即用一个小惯性环节来等效地替代多个小惯性环节，等效的小惯性环节的时间常数等于被替代的多个小惯性环节的时间常数之和。

系统开环对数幅频特性在高频段的幅值，直接反映了系统对高频干扰信号的抑制能力。高频部分的幅值越低，系统的抗干扰能力越强。

总之，为了使系统满足一定的稳态和动态要求，对开环对数幅频特性的形状有如下要求：低频段要有一定的高度和斜率；中频段的斜率最好为 - 20dB/dec，且具有足够的宽度；高频段采用迅速衰减的特性，以抑制不必要的高频干扰。

7.3.1.2　闭环频率特性与时域响应的关系

对于二阶系统，其频域性能指标与时域性能指标之间存在一定的数学关系。

二阶系统的闭环传递函数为：

$$\phi(s)=\frac{\omega_n^{\ 2}}{s^2+2\zeta\omega_n s+\omega_n^{\ 2}} \tag{7-14}$$

系统的闭环频率特性为：

$$\phi(\mathrm{j}\omega)=\frac{\omega_n^{\ 2}}{(\mathrm{j}\omega)^2+\mathrm{j}2\zeta\omega_n\omega+\omega_n^{\ 2}} \tag{7-15}$$

系统的闭环幅频特性为：

$$M(\omega)=\frac{\omega_n^{\ 2}}{\sqrt{(\omega_n^{\ 2}-\omega^2)^2+(2\zeta\omega_n\omega)^2}} \tag{7-16}$$

系统的闭环相频特性为：

$$\varphi(\omega)=-\mathrm{arctg}\frac{2\zeta\omega_n\omega}{\omega_n^{\ 2}-\omega^2} \tag{7-17}$$

二阶系统的谐振峰值 M_r 与时域超调量 M_p 之间的关系为：

$$M_p=\mathrm{e}^{-\zeta\pi/\sqrt{1-\zeta^2}}\times100\% \tag{7-18}$$

$$M_r=\frac{1}{2\zeta\sqrt{1-\zeta^2}} \tag{7-19}$$

从上面的式子可以看出：

（1）谐振峰值 M_r 仅与阻尼比 ζ 有关，超调量 M_p 也仅取决于阻尼比 ζ；

（2）ζ 越小，M_r 增加得越快，此时超调量 M_p 会很大，超过 40%，这样的系统一般不符合瞬态响应指标的要求；

（3）当 $0.4<\zeta<0.707$ 时，M_r 与 M_p 的变化趋势基本一致，此时谐振峰值 $M_r=1.2\sim1.5$，超调量 $M_p=20\%\sim30\%$，系统响应结果比较理想；

（4）当 $\zeta>0.707$ 时，无谐振峰值，M_r 与 M_p 的对应关系不再存在，通常在设计中 ζ 取值在 0.4 至 0.7 之间。

二阶系统的谐振频率 ω_r 与峰值时间 t_p 之间的关系为：

$$t_p\omega_r=\frac{\pi\sqrt{1-2\zeta^2}}{\sqrt{1-\zeta^2}} \tag{7-20}$$

从上式可以看出，当ζ为常数时，谐振频率ω_r与峰值时间t_p成反比，ω_r值越大，t_p越小，表示系统时间响应越快。

二阶系统的闭环截止频率ω_b与过渡过程时间t_s之间的关系为：

$$\omega_b t_s = \frac{3 \sim 4}{\zeta}\sqrt{1-2\zeta^2+\sqrt{2-4\zeta^2+4\zeta^4}} \tag{7-21}$$

从上式可以看出，当阻尼比ζ给定后，闭环截止频率ω_b与过渡过程时间t_s成反比。换言之，ω_b越大（频带宽度越宽），系统的响应速度越快。

【例 7-4】 已知二阶系统的传递函数：

$$G(s)=\frac{3.6}{s^2+3s+5}$$

试计算此系统的谐振幅值和谐振频率。

解：计算谐振幅值和谐振频率的 MATLAB 程序代码如下。

```
function [Mr, Pr, Wr]=mr(G)
[mag, pha, w]=bode(G)
magn(1, :)=mag(1, :)
phase(1, :)=pha(1, :)
[M, i]=max(magn)
Mr=20*log10(M)
Pr=phase(1, i)
Wr=w(i, 1)
```

主函数的 MATLAB 程序代码如下。

```
num=[3.6]
den=[1, 3, 5]
G=tf(num, den)
[Mr, Pr, Wr]=mr(G)
```

程序运行结果如下：

```
Mr =
    -2.8098
Pr =
  -24.6446
Wr =
     0.6915
```

由运行结果可知，系统的谐振峰值M_r=−2.8098dB，谐振频率ω_r=0.6915rad/s。

还可以从 MATLAB 绘制的频率响应曲线上直接得到谐振峰值和谐振频率。步骤是先生成频率响应曲线，然后在频率响应图内部空白处用鼠标右键单击，弹出菜单，选择“Peak Response”菜单项，将在频率响应图上出现一个圆点，该点就是系统的谐振频率处。

例 7-4 的绘制 Bode 图的 MATLAB 程序代码如下：

```
num=[3.6]
```

```
den=[1, 3, 5]
G=tf(num, den)
bode(G)
```

程序输出如图 7.10 所示的 Bode 图（图中的弹出菜单非 Bode 图内容，作用见下）。

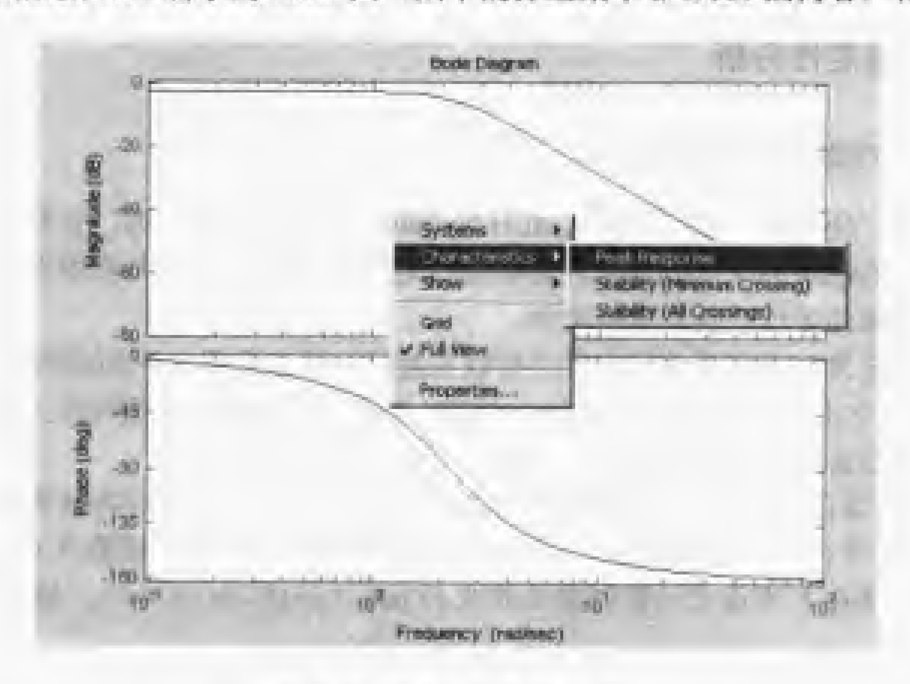


图 7.10　例 7-4 系统 Bode 图

单击鼠标右键，弹出菜单，选择“Peak Response”菜单项，将光标移至圆点处，输出如图 7.11 所示的图形。

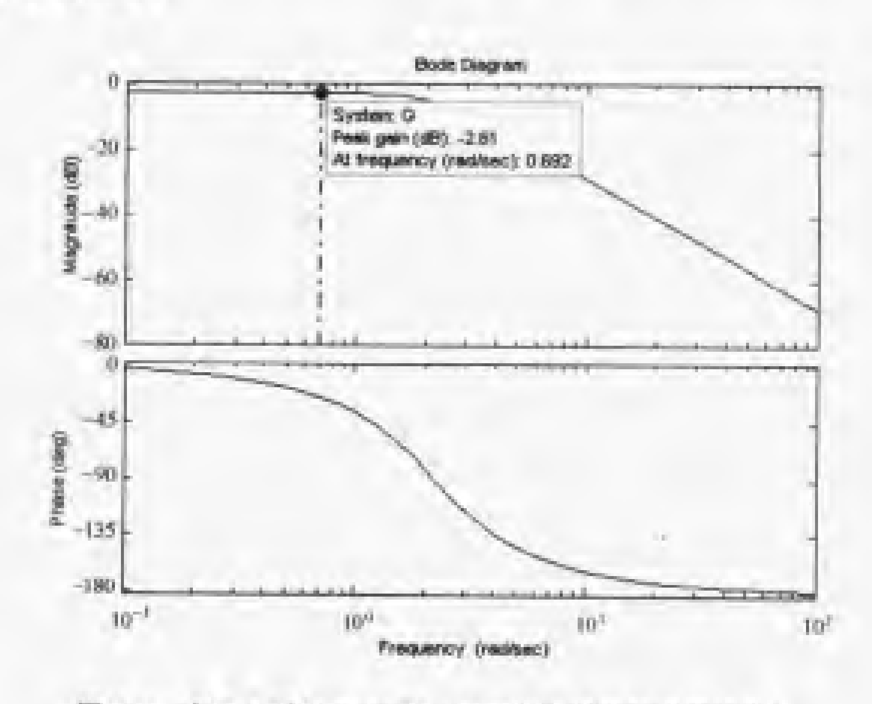


图 7.11　例 7-4 系统 Bode 图（显示谐振峰值和谐振频率）

从图中可以看出，系统的谐振峰值 M_r =−2.81dB，谐振频率 ω_r =0.692rad/s，与前面的计算结果是一致的。

也可以用同样的方法直接从 MATLAB 绘制的 Nyquist 图和 Nichols 图中得到谐振峰值和谐振频率。

7.3.2　稳定性分析

7.3.2.1　乃奎斯特稳定判据

乃奎斯特（Nyquist）稳定判据是由 H.Nyquist 于 1932 年提出的。它利用开环系统幅相频率特性（Nyquist 图）来判断闭环系统的稳定性。Nyquist 稳定判据的理论基础是复变函数理论中的幅角定理，也称映射定理。

为将映射定理与控制系统稳定性分析联系起来，适当选择 S 平面的封闭曲线 C_s，如图 7.12 所示。它由整个虚轴和半径为 ∞ 的右半圆组成，试验点按顺时针方向移动一圈，得到的封闭曲线称为 Nyquist 轨迹。

Nyquist 轨迹在 $F(s)$ 平面上的映射也是一条封闭曲线，称为 Nyquist 曲线。

Nyquist 轨迹 C_s 由两部分组成，一部分沿虚轴由下而上移动，试验点 $s=\mathrm{j}\omega$ 在整个虚轴上的移动，其在 F 平面上的映射就是曲线 $F(\mathrm{j}\omega)$（ω 由 $-\infty \to +\infty$），如图 7.13 所示。

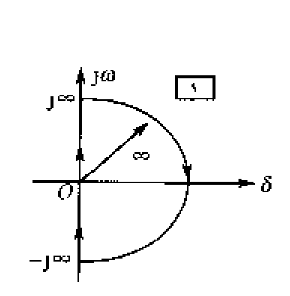


图 7.12　S 平面上的 Nyquist 轨迹

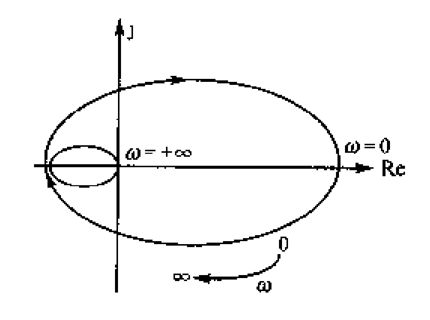


图 7.13　F 平面上的 Nyquist 曲线

曲线也可用式（7-22）表示：

$$F(\mathrm{j}\omega)=1+G(\mathrm{j}\omega)H(\mathrm{j}\omega) \tag{7-22}$$

Nyquist 轨迹 C_s 的另一部分为 S 平面上半径为 ∞ 的右半圆，映射到 F 平面上为：

$$F(\infty)=1+G(\infty)H(\infty) \tag{7-23}$$

根据映射定理可得，S 平面上的 Nyquist 轨迹在 F 平面上的映射 $F(\mathrm{j}\omega)$（ω 由 $-\infty \to +\infty$）逆时针包围坐标原点的次数 N 为：

$$N=P-Z \tag{7-24}$$

式中，Z 表示位于右半平面 $F(s)=1+G(s)H(s)$ 的零点数，即闭环右极点个数；

P 表示位于右半平面 $F(s)=1+G(s)H(s)$ 的极点数，即开环右极点个数；

N 表示 Nyquist 曲线包围坐标原点的次数。

闭环系统稳定的条件为系统的闭环极点均在 S 平面的左半平面，即 Z=0 或 N=P。

1. Nyquist 稳定判据一

当系统开环传递函数 $G(s)H(s)$ 在 s 平面的原点及虚轴上无极点时，Nyquist 稳定判据可表示为：当 ω 从 $-\infty \to +\infty$ 变化时的 Nyquist 曲线 $G(j\omega)H(j\omega)$ 逆时针包围（−1, j0）点的次数 N 等于系统 $G(s)H(s)$ 位于右半 S 平面的极点数 P 时，即当 $N=P$ 时，闭环系统稳定，否则闭环系统不稳定（$N \neq P$）。

由 Nyquist 曲线 $G(j\omega)H(j\omega)$（ω 从 $0 \to +\infty$）判别闭环系统稳定性的 Nyquist 判据为：$G(j\omega)H(j\omega)$ 曲线（ω 为 $0 \to +\infty$）逆时针包围(−1, j0)的次数为 $\frac{P}{2}$。

2. Nyquist 稳定判据二

设系统开环传递函数为：

$$G(s)H(s)=\frac{K\prod_{j=1}^{m}(\tau_j s+1)}{s^{\nu}\prod_{i=1}^{n-\nu}(T_i s+1)} \tag{7-25}$$

式中，ν 表示开环传递函数中位于原点的极点个数。

Nyquist 轨迹的修正如图 7.14 所示。

Nyquist 轨迹由 4 部分组成：

- 以原点为圆心，以无限大为半径的大半圆；
- 由 $-j\infty$ 到 $j0_-$ 的负虚轴；
- 由 $j0_+$ 沿正虚轴到 $+j\infty$；
- 以原点圆心，以 ε（$\varepsilon \to 0$）为半径的从 $j0_-$ 到 $j0_+$ 的小半圆。

考虑 S 平面上有位于坐标原点的 ν 个极点，Nyquist 曲线稳定判据为：当系统开环传递函数有 ν 个极点位于 S 平面坐标原点时，如果增补开环频率特性曲线 $G(j\omega)H(j\omega)$（ω 从 $-\infty \to +\infty$）逆时针包围（−1，j0）点的次数 N 等于系统开环右极点个数 P，则闭环系统稳定，否则系统不稳定。

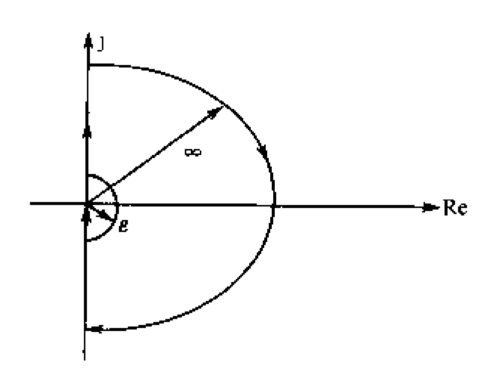


图 7.14　绕过原点的 Nyquist 轨迹

Nyquist 对数稳定判据是对数幅相频率特性的稳定判据，实际上是 Nyquist 稳定判据的另一种形式，即利用开环系统的对数频率特性曲线（Bode 图）来判别闭环系统的稳定性，而 Bode 图又可通过试验获得，因此在工程上获得了广泛的应用。

Nyquist 图与 Bode 图的对应关系如图 7.15 所示。

采用对数频率特性曲线（Bode 图）时，Nyquist 稳定判据可表述为：当 ω 由 $-\infty \to +\infty$ 变化时，在开环对数幅频特性曲线 $L(\omega) \geqslant 0$ 的频段内，相频特性曲线对−180° 线的正穿越与负穿越次数之差为 P（P 为 S 平面右半平面开环极点数），则闭环系统稳定。

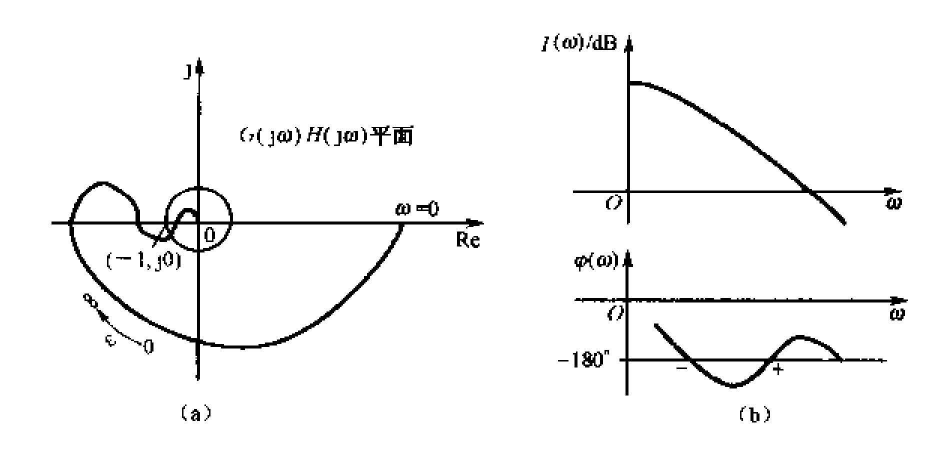


图 7 15　Nyquist 图和 Bode 图的对应关系

7.3.2.2　稳定裕度

1. 相对稳定性

从 Nyquist 稳定判据可知，若系统开环传递函数没有右半平面的极点且闭环系统是稳定的，则开环系统的 Nyquist 曲线离（−1, j0）点越远，闭环系统的稳定性越好；开环系统的 Nyquist 曲线离（−1, j0）点越近，闭环系统的稳定性越差，这就是通常所说的控制系统相对稳定性。

系统的相对稳定性用 Nyquist 曲线相对点（−1, j0）的靠近程度来度量，定量表示为增益裕度和相角裕度。

2. 增益裕度

增益裕度用于表示 $G(\mathrm{j}\omega)H(\mathrm{j}\omega)$ 曲线在负实轴上相对于（−1, j0）点的靠近程度。

当 $G(\mathrm{j}\omega)H(\mathrm{j}\omega)$ 曲线与负实轴交于 G 点时，G 点的频率 ω_g 称为相位穿越越率，此时 ω_g 处的相角为−180°，幅值为 $|G(\mathrm{j}\omega)H(\mathrm{j}\omega)|$，开环频率特性幅值 $|G(\mathrm{j}\omega)H(\mathrm{j}\omega)|$ 的倒数称为增益裕度（或幅值裕度），用 K_g 表示：

$$K_g=\frac{1}{|G(\mathrm{j}\omega_g)H(\mathrm{j}\omega_g)|} \tag{7-26}$$

式中，ω_g 满足：

$$\angle G(\mathrm{j}\omega_g)H(\mathrm{j}\omega_g)=-180^\circ \tag{7-27}$$

图 7.16 中的 ω_g 在 Bode 图中对应于相越特性上相角为−180° 的越率，如图 7.17 所示。增益裕度用分贝数表示为：

$$K_g=-20\lg|G(\mathrm{j}\omega_g)H(\mathrm{j}\omega_g)| \quad (\mathrm{dB}) \tag{7-28}$$

对于最小相位系统：

- 当 $|G(\mathrm{j}\omega_g)H(\mathrm{j}\omega_g)|<1$ 或 $20\lg|G(\mathrm{j}\omega_g)H(\mathrm{j}\omega_g)|<0$ 时，闭环系统稳定；

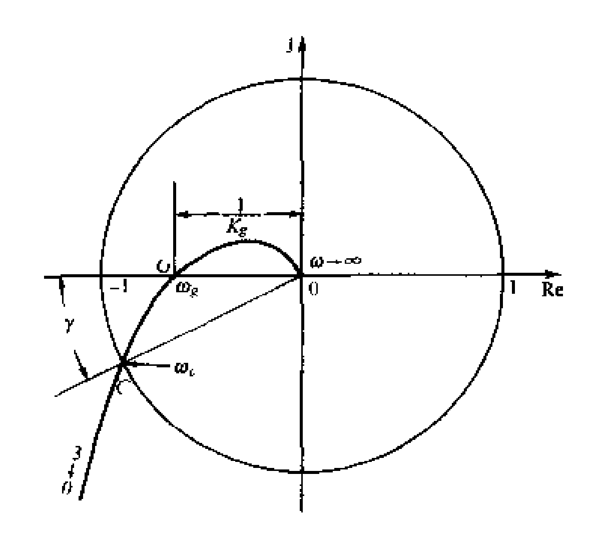


图 7.16　最小相位系统的 Nyquist 图

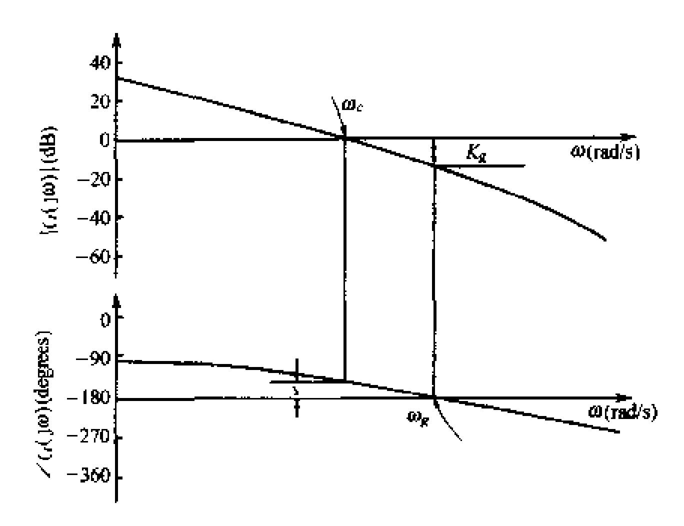


图 7.17　对数频率特性

- 当$\left|G(j\omega_g)H(j\omega_g)\right|>1$或$20\lg\left|G(j\omega_g)H(j\omega_g)\right|>0$时，闭环系统不稳定；
- 当$\left|G(j\omega_g)H(j\omega_g)\right|=1$或$20\lg\left|G(j\omega_g)H(j\omega_g)\right|=0$时，系统处于临界状态。

临界状态下的开环系统不稳定。为使临界状态下的闭环系统稳定，$G(j\omega)H(j\omega)$曲线应包围（$-1, j0$）点，此时$K_g=-20\lg\left|G(j\omega_g)H(j\omega_g)\right|<0$，闭环系统稳定。

因此，增益裕度K_g表示系统处于临界状态时，系统增益所允许的增大倍数。

3. 相角裕度

为了表示系统相角变化对系统稳定性的影响，引入相角裕度的概念。

ω_c称为穿越频率，也称剪切频率或截止频率，在图 7.16 中，$G(j\omega)H(j\omega)$与单位圆相交于 C 点，C 点处的频率为ω_C，此时$\left|G(j\omega)H(j\omega)\right|=1$。

使系统达到临界稳定状态而尚可增加的滞后相角就称为系统的相角裕度或相角裕度，表示为：

$$\gamma=180^\circ+\psi(\omega_c) \tag{7-29}$$

相角裕度γ是增益穿越频率ω_C处相角与-180°线之间的距离。

对于最小相位系统：

- 当$\gamma>0$时，闭环系统稳定；
- 当$\gamma<0$时，闭环系统不稳定。

增益裕度和相角裕度通常作为设计和分析控制系统的频域指标，但仅用其中之一是不足以说明系统相对稳定性的。

4. 用幅相频率特性曲线分析系统稳定性

若采用幅相频率特性曲线，如果当$G(j\omega)H(j\omega)$的开环增益发生变化时，曲线仅是上下简单平移，而当对$G(j\omega)H(j\omega)$增加一个恒定相角时，曲线为水平平移，那么这对分析系统稳定性与系统参数之间的相互影响是非常有利的。

7.3.2.3　MATLAB 在稳定性分析中的应用

MATLAB 提供了用于计算系统稳定裕度的函数 margin，其调用法则如下。

margin 函数可以从频率响应数据中计算出幅值裕度、相角裕度以及对应的频率。幅值裕度和相角裕度是针对开环 SISO 系统而言的，它指出了系统在闭环时的相对稳定性。当不带输出变量引用时，margin 可在当前图形窗口中绘出带有裕量及相应频率显示的 Bode 图，其中的幅值裕度以分贝为单位。

幅值裕度是在相角为-180° 处使开环增益为 1 的增益量，如在-180° 相频处的开环增益为 g，则幅值裕度为 1/g；若用分贝值表示幅值裕度，则为-20lg10(g)。类似地，相角裕度是当开环增益为 1.0 时，相应的相角与 180° 角的和。

margin(mag, phase, w)：由 bode 指令得到的幅值 mag（不是以 dB 为单位）、相角 phase 及角频率 w 矢量绘出带有裕量及相应频率显示的 Bode 图。

margin(num, den)：计算连续系统传递函数表示的幅值裕度和相角裕度，并绘出相应的 Bode 图。类似地，margin(a, b, c, d)可计算出连续状态空间系统表示的幅值裕度和相角裕度，并绘出相应的 Bode 图。

[gm, pm, wcg, wcp]＝margin(mag, phase, w)：由幅值 mag（不是以 dB 为单位）、相角 phase 及角频率 w 矢量计算系统幅值裕度和相角裕度及相应的相角交界频率 wcg、截止频率 wcp，而不直接绘出 Bode 图。

MATLAB 6.0 以上版本还提供了计算系统稳定裕度的函数 allmargin，其调用法则如下。

s＝allmargin(sys)：计算幅值裕度、相角裕度以及对应的频率。幅值裕度和相角裕度是针对开环 SISO 系统而言，输出 s 是一个结构体，它包括幅值裕度、相角裕度以及对应的频率、时滞增益裕度。

【例 7-5】　已知一高阶系统的开环传递函数为：

$$G(s)=\frac{5(0.0167s+1)}{s(0.03s+1)(0.0025s+1)(0.001s+1)}$$

试计算系统的相角稳定裕量和幅值稳定裕量，并绘制系统的 Bode 图。

解：MATLAB 程序代码如下。

```
num=5*[0.0167, 1]
den=conv( conv([1, 0], [0.03, 1]), conv([0.0025, 1], [0.001, 1]))
G=tf(num, den)
w=logspace(0, 4, 50)
bode(G, w)
grid
[Gm, Pm, Wcg, Wcp]=margin(G)
```

响应曲线如图 7.18 所示。

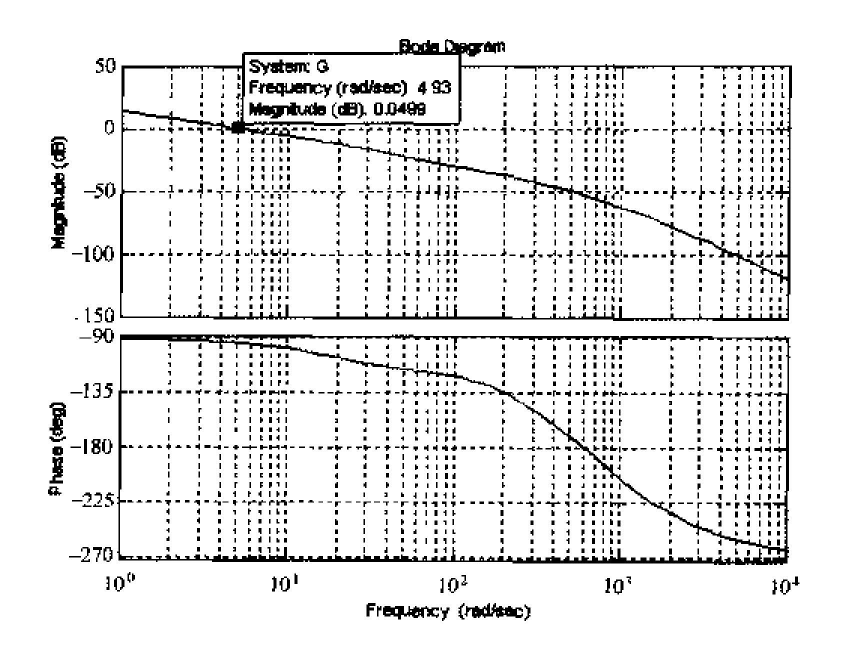


图 7.18　例 7-5 系统的 Bode 图

运行结果如下：

```
Gm =
  455.2548
Pm =
   85.2751
Wcg =
  602.4232
Wcp =
    4.9620
```

由系统运行结果可知，幅值稳定裕量 $K_g=455.2548$；相角稳定裕量 $\gamma=85.2751°$；相角穿越频率 $\omega_g=602.4232$ rad/s；幅值穿越频率（截止频率）$\omega_c=4.9620$ rad/s 。

【例 7-6】　已知一高阶系统的开环传递函数为：

$$G(s)=\frac{K(0.0167s+1)}{s(0.03s+1)(0.0025s+1)(0.001s+1)}$$

试计算当开环增益 $K=5,\ 500,\ 800,\ 3000$ 时，系统稳定裕量的变化。

解：MATLAB 程序代码如下。

```
k=[5, 500, 800, 3000]
for j=1:4
num=k(j)*[0.0167, 1]
den=conv( conv([1, 0], [0.03, 1]), conv([0.0025, 1], [0.001, 1]))
G=tf(num, den)
y(j)－allmargin(G)
```

```
end
y(1)
y(2)
y(3)
y(4)
```

运行结果如下：

```
%y(1)
ans =
    GMFrequency: 602.4232
     GainMargin: 455.2548
    PMFrequency: 4.9620
    PhaseMargin: 85.2751
    DMFrequency: 4.9620
    DelayMargin: 0.2999
         Stable: 1
%y(2)
ans =
    GMFrequency: 602.4232
     GainMargin: 4.5525
    PMFrequency: 237.7216
    PhaseMargin: 39.7483
    DMFrequency: 237.7216
    DelayMargin: 0.0029
         Stable: 1
%y(3)
ans =
    GMFrequency: 602.4232
     GainMargin: 2.8453
    PMFrequency: 329.9063
    PhaseMargin: 27.7092
    DMFrequency: 329.9063
    DelayMargin: 0.0015
         Stable: 1
%y(4)
ans =
    GMFrequency: 602.4232
     GainMargin: 0.7588
    PMPrequency: 690.5172
    PhaseMargin: -6.7355
```

```
DMFrequency: 690.5172
DelayMargin: 0.0089
    Stable: 0
```

由系统运行结果可知，随着开环增益的增大，相角稳定裕度在减小，表明系统的稳定性在变差。当 $K=3000$ 时，相角稳定裕度变为负值，此时系统不稳定了。

【例 7-7】 已知系统的开环传递函数为：

$$GH(s)=\frac{100(s+5)}{(s-2)(s+8)(s+20)}$$

试绘制系统的极坐标图，并利用 Nyquist 稳定判据判断闭环系统的稳定性。

解：MATLAB 程序代码如下。

```
k=100
z=[-5]
p=[2, -8, -20]
G=zpk(z, p, k)
nyquist(G)
grid
```

运行结果如图 7.19 所示。

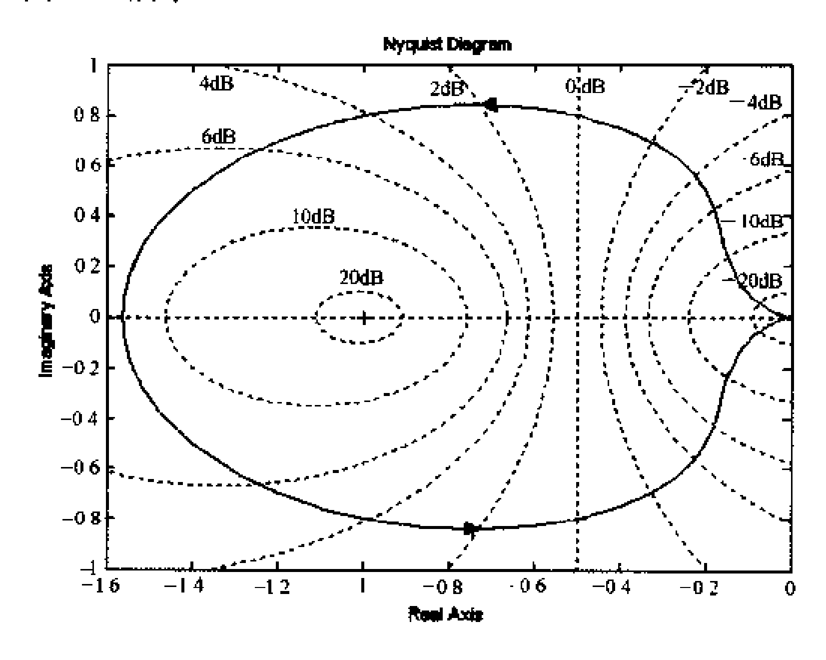


图 7.19　例 7-7 开环系统的 Nyquist 图

从图 7.19 可以看出，开环系统有一个右半 S 平面的极点（$p=1$），因此开环系统是不稳定的。开环系统的 Nyquist 图逆时针包围（$-1, j0$）点 1 次，那么根据 Nyquist 稳定判据可知，闭环系统是稳定的。

【例 7-8】 已知系统的开环传递函数为：

$$GH(s)=\frac{100K}{s(s+5)(s+10)}$$

试分别绘制 $K=1, 7.8, 20$ 时系统的极坐标图，并利用 Nyquist 稳定判据判断闭环系统的稳定性。

解：MATLAB 程序代码如下。

```
clear
z=[]
p=[0, -5, -10]
k=100.*[1, 7.8, 20]
G=zpk(z, p, k(1))
[re1, im1]=nyquist(G)
G=zpk(z, p, k(2))
[re2, im2]=nyquist(G)
G=zpk(z, p, k(3))
[re3, im3]=nyquist(G)
plot( re1(:), im1(:), re2(:), im2(:), re3(:), im3(:))
v=[-5, 1, -5, 1];
axis(v)
grid
xlabel('Real Axis')
ylabel('Imaginary Axis')
text(-0.4, -3.6, 'K=1')
text(-2.7, -2.7, 'K=7.8')
text(-4.4, -1.6, 'K=20')
```

运行结果如图 7.20 所示。

该开环系统没有右半 S 平面的极点（$p=0$），因此开环系统是稳定的。从图 7.20 可以看出，当 $K=1$ 时，开环系统的 Nyquist 图不包围（-1, j0）点，根据 Nyquist 稳定判据，系统是稳定的。当 $K=7.8$ 和 $K=20$ 时，开环系统的 Nyquist 图包围（-1, j0）点，根据 Nyquist 稳定判据，系统是不稳定的。

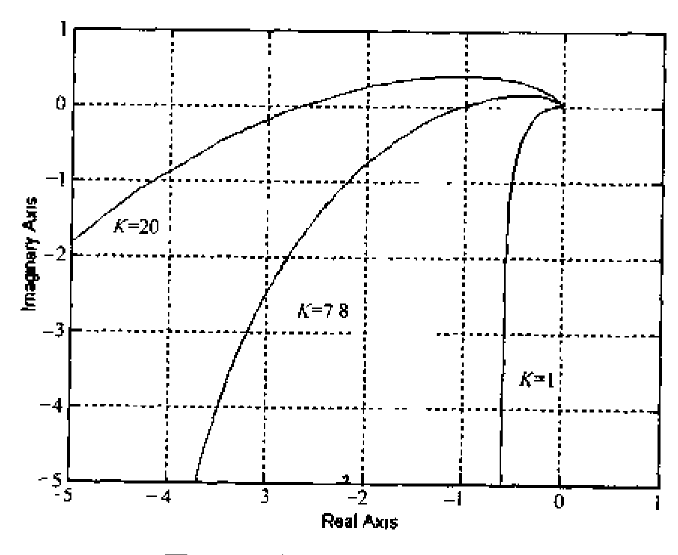


图 7.20　例 7-8 的 Nyquist 图

第 8 章　控制系统校正与综合

8.1　引　　言

在实际工程控制中，往往需要设计一个系统并选择适当的参数以满足性能指标的要求，或对原有系统增加某些必要的元件或环节，使系统能够全面满足性能指标要求，此类问题就称为系统校正与综合，或称为系统设计。系统设计过程是一个反复试探的过程，需要许多经验的积累。MATLAB/Simulink 为系统设计提供了有效手段。

本章介绍控制系统校正与综合的基本概念和常用方法，重点阐述 PID 控制器的设计原理，以及基于 MATLAB/Simulink 的线性控制系统设计方法。

通过本章，读者对控制系统校正与综合的基本概念和基本方法能有比较全面的认识，并学会运用 MATLAB/Simulink 对控制系统进行设计。

8.2　控制系统校正与综合基础

设计控制系统的目的是使控制系统满足特定的性能指标，性能指标与控制精度、相对稳定性、响应速度等因素有关。在设计控制系统时，确定控制系统性能指标是非常重要的工作。

8.2.1　控制系统性能指标

性能指标有多种形式，不同的设计方法选用的性能指标是不同的，不同的性能指标之间又存在着某些联系，这些都需要在确定性能指标时仔细考虑。

8.2.1.1　性能指标概述

按类型，控制系统的性能指标可分为：

- 时域性能指标，包括稳态性能指标和动态性能指标；
- 频域性能指标，包括开环频域指标和闭环频域指标。

性能指标分类如图 8.1 所示。

在控制系统设计中，采用的设计方法一般依据性能指标的形式而定。如果性能指标以单位阶跃响应的峰值时间、调节时间等时域特征量给出，那么一般采用根轨迹法进行设计；如果性能指标以相角裕度、幅值裕度等频域特征量给出，那么一般采用频率法进行设计。工程上通常采用频率法进行设计，因此需要通过近似公式对时域和频域两种性能指标进行转换。

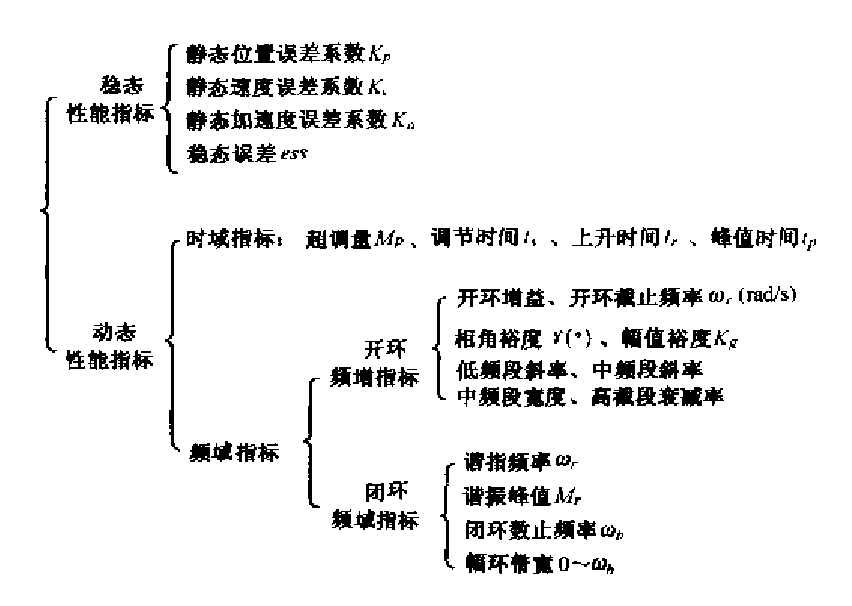


图 8.1　控制系统性能指标分类图

8.2.1.2　二阶系统频域指标与时域指标的关系

各类性能指标是从不同的角度表示系统性能的，它们之间存在内在联系。二阶系统是设计中最常见的系统，对于二阶系统，时域指标和频域指标能用数学公式准确地表示出来，它们可统一采用阻尼比ζ和无阻尼自然振荡频率ω_n来进行如下描述。

（1）超调量：$M_p = e^{\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} \times 100\%$

（2）调节时间：$t_s = \dfrac{3.5}{\zeta\omega_n}$，$\omega_c t_s = \dfrac{7}{\mathrm{tg}\gamma}$

（3）上升时间：$t_r = \dfrac{\pi - \mathrm{arctg}\dfrac{\sqrt{1-\zeta^2}}{\zeta}}{\omega_n\sqrt{1-\zeta^2}}$

（4）谐振峰值：$M_r = \dfrac{1}{2\zeta\sqrt{1-\zeta^2}}$，$0 \leqslant \zeta \leqslant \dfrac{\sqrt{2}}{2}$

（5）谐振频率：$\omega_r = \omega_n\sqrt{1-2\zeta^2}$

（6）带宽频率：$\omega_b = \omega_n\sqrt{1-2\zeta^2+\sqrt{2-4\zeta^2+2\zeta^4}}$

（7）相位裕度：$\gamma = \mathrm{arctg}\dfrac{2\zeta}{\sqrt{\sqrt{1+4\zeta^4}-2\zeta^2}}$

（8）截止频率：$\omega_c = \omega_n\sqrt{\sqrt{1+4\zeta^4}-2\zeta^2}$

8.2.2　控制系统校正概述

为使控制系统能满足一定的性能指标，通常需要在控制系统中引入一定的附加装置，

称为控制器或校正装置。

根据校正装置的特性，可分为超前校正装置、滞后校正装置和滞后—超前校正装置。

1. 超前校正装置

校正装置输出信号在相位上超前于输入信号，即校正装置具有正的相角特性，这种校正装置称为超前校正装置，对系统的校正称为超前校正。

2. 滞后校正装置

校正装置输出信号在相位上滞后于输入信号，即校正装置具有负的相角特性，这种校正装置称为滞后校正装置，对系统的校正称为滞后校正。

3. 滞后—超前校正装置

校正装置在某一频率范围内具有负的相角特性，而在另一频率范围内却具有正的相角特性，这种校正装置称为滞后—超前校正装置，对系统的校正称为滞后—超前校正。

根据校正装置与被控对象的不同连接方式，可分为串联校正、反馈（并联）校正、前馈校正和干扰补偿等。串联校正和并联校正是最常见的两种校止方式。

4. 串联校正

如果校正元件与系统的不可变部分串联起来，如图 8.2 所示，则称这种形式的校正为串联校正。串联校正通常设置在前向通道中能量较低的点，为此通常需要附加放大器以增大增益，补偿校正装置的衰减或进行隔离。

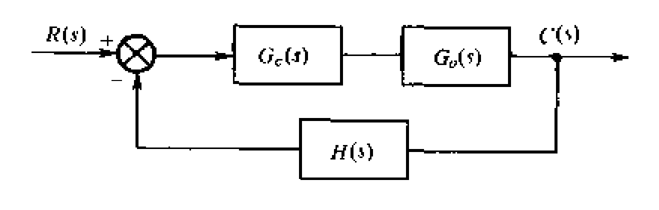


图 8.2　串联校正

图中的 $G_0(s)$ 表示前向通道不可变部分的传递函数，$H(s)$ 表示反馈通道不可变部分的传递函数，$G_c(s)$ 表示校正部分的传递函数。

5. 反馈校正

如果从系统的某个元件输出取得反馈信号，构成反馈回超，并在反馈回路内设置传递函数为 $G_c(s)$ 的校正元件，如图 8.3 所示，则称这种形式的校正为反馈校正。反馈削弱了前向超道上元件变化的影响，具有较离的灵敏度，单位反馈时也容易控制偏馈，这就是较多地采用反馈校正的原因。

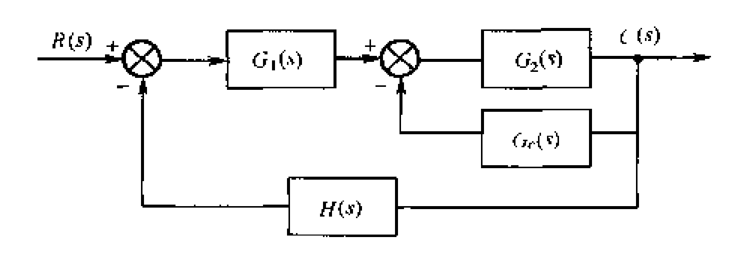


图 8.3　反馈校正

6. 前馈校正

如果从系统的输入元件输出取得前馈信号，构成前馈回路，并在前馈回路内设置传递函数为 $G_c(s)$ 的校正元件，如图 8.4 所示，则称这种形式的校正为前馈校正。它是在系统反馈回路之外采用的校正方式之一。前馈校正通常用于补偿系统外部扰动的影响，也可用于对控制输入进行校正。

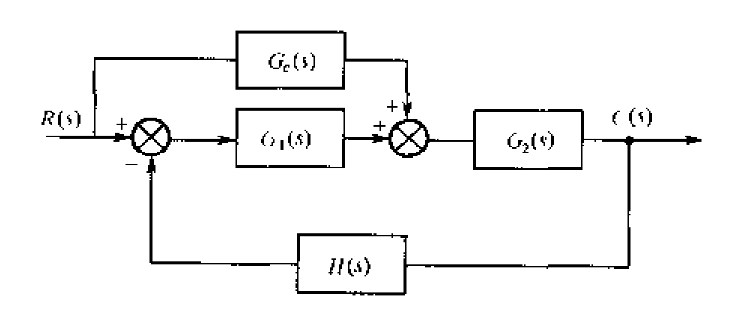


图 8.4　前馈校正

8.3　PID 控制器设计

8.3.1　PID 控制器概述

当今的自动控制校术大部分是基于反馈概念的。反馈理论包括三个基本要素：测量、比较和执行。测量关心的是变量，并与期望值相比较，以此误差来纠正和调节控制系统的响应。反馈理论及其在自动控制中应用的关键是做出正确测量与比较后，如何用于系统的纠正与调节。

在过去的几十年里，PID 控制器在工业控制中得到了广泛应用。在控制理论和技术飞速发展的今天，工业过程控制中 95%以上的控制回路都具有 PID 结构，并且许多高级校制都是以 PID 控制为基础的。

PID（比例—积分—微分）控制器作为最早实用化的控制器已有 70 多年历史，现在仍然是应用最广泛的工业控制器。PID 控制器简单易懂，使用中不需要精确的系统模型等先决条件，因而成为应用最为广泛的校制器。

PID 控制器由比例单元（P）、积分单元（I）和微分单元（D）组成。在校制系统的

设计与校正中，PID 控制规律的优越性是明显的，它的基本原理却比较简单。

基本的 PID 控制规律可描述为：

$$G_c(s) = K_P + \frac{K_I}{s} + K_D s \tag{8-1}$$

PID 控制器由于用途广泛，使用灵活，已有系列化产品，使用中只需设定三个参数（K_P，K_I 和 K_D）即可。在很多情况下，并不一定需要三个单元，可以取其中的一或两个单元，不过比例控制单元是必不可少的。

PID 控制器具有以下优点：

（1）原理简单，使用方便，PID 参数（K_P，K_I 和 K_D）可以根据过程动态特性及时调整。如果过程的动态特性发生变化，如对负载变化引起的系统动态特性变化，PID 参数就可以重新进行调整与设定。

（2）适应性强，按 PID 控制规律进行工作的控制器早已商品化，即使目前最新式的过程控制计算机，其基本控制功能也仍然是 PID 控制。PID 应用范围广，虽然很多工业过程是非线性或时变的，但通过适当简化，可以将其变成基本线性和动态特性不随时间变化的系统，这样就可以通过 PID 控制了。

（3）鲁棒性强，即其控制品质对被控制对象特性的变化不太敏感。

PID 也有其固有的缺点。PID 在控制非线性、时变、耦合及参数和结构不确定的复杂过程时，效果不是太好；最主要的是，如果 PID 控制器不能控制复杂过程，无论怎么调参数都没用。

虽然有这些缺点，在科学技术尤其是计算机迅速发展的今天，虽说涌现出了许多新的控制方法，但 PID 仍因其自身的优点而得到了最广泛的应用，PID 控制规律仍是最普遍的控制规律。PID 控制器是最简单但许多时候仍最好的控制器。

8.3.2　比例（P）控制

比例控制是一种最简单的控制方式。其控制器的输出与输入误差信号成比例关系。当仅有比例控制时系统输出存在稳态误差。比例控制器的传递函数为：

$$G_c(s) = K_P \tag{8-2}$$

式中，K_P 称为比例系数或增益（视情况可设置为正或负），一些传统的控制器又常用比例带（Proportional Band，PB）来取代比例系数 K_P，比例带是比例系数的倒数，比例带也称为比例度。

对于单位反馈系统，0 型系统响应实际阶跃信号 $R_0 1(t)$ 的稳态误差与其开环增益 K 近似成反比，即

$$\lim_{t\to\infty} e(t) = \frac{R_0}{1+K} \tag{8-3}$$

对于单位反馈系统，I 型系统响应匀速信号 $R_1 t$ 的稳态误差与其开环增益 K_v 近似成反比，即

$$\lim_{t\to\infty} e(t) = \frac{R_1}{K_v} \tag{8-4}$$

P 控制只改变系统的增益而不影响相位，它对系统的影响主要反映在系统的稳态误差和稳定性上，增大比例系数可提高系统的开环增益，减小系统的稳态误差，从而提高系统的控制精度，但这会降低系统的相对稳定性，甚至可能造成闭环系统的不稳定，因此，在系统校正和设计中，P 控制一般不单独使用。

具有比例控制器的系统结构如图 8.5 所示。

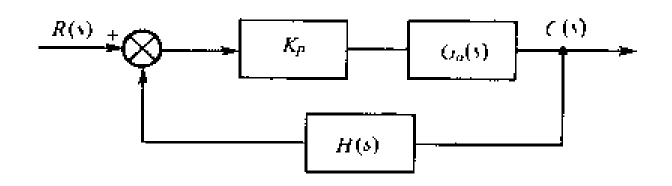


图 8.5　具有比例控制器的系统结构图

系统的特征方程为：

$$D(s)=1+K_pG_o(s)H(s)=0 \qquad (8\text{-}5)$$

下面的例子给出了一个直观的概念，用以说明纯比例控制的作用或比例调节对系统性能的影响。

【例 8-1】　控制系统如图 8.5 所示，其中 $G_o(s)$ 为三阶对象模型：

$$G_o(s)=\frac{1}{(s+1)(s+2)(s+5)}$$

$H(s)$ 为单位反馈，对系统单采用比例控制，比例系数分别为 $K_p=0.1, 2.0, 2.4, 3.0, 3.5$，试求各比例系数下系统的单位阶跃响应，并绘制响应曲线。

解：MATLAB 程序代码如下。

```
G=tf(1, conv( conv([1, 1], [2, 1]), [5, 1]) );
kp=[0.1, 2.0, 2.4, 3.0, 3.5]
for i=1:5
G=feedback(kp(i)*G, 1);
step(G)
hold on
end
gtext('kp=0.1')
gtext('kp=2.0')
gtext('kp=2.4')
gtext('kp=3.0')
gtext('kp=3.5')
```

响应曲线如图 8.6 所示。

从图 8.6 可以看出，随着 K_p 值的增大，系统响应速度也加快，系统的超调也随着增加，调节时间也随着增长。但当 K_p 增大到一定值后，闭环系统将趋于不稳定。

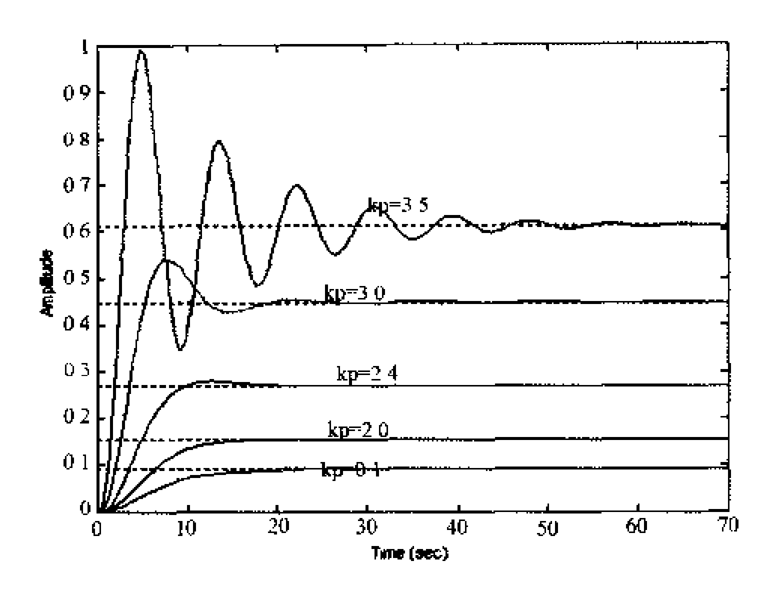


图 8.6　例 8-1 系统阶跃响应图

8.3.3　比例微分（PD）控制

具有比例加微分控制规律的控制称为 PD 控制，PD 的传递函数为。

$$G_c(s) = K_P + K_P \tau s \tag{8-6}$$

式中，K_P 为比例系数，τ 为微分时间常数，K_P 与 τ 两者都是可调的参数。

具有 PD 控制器的系统结构如图 8.7 所示。

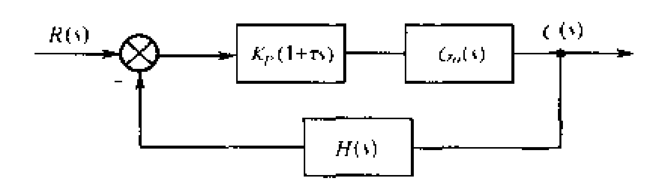


图 8.7　具有比例微分控制器的系统结构图

PD 控制器的输出信号为：

$$u(t) = K_P e(t) + K_P \tau \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{8-7}$$

在微分控制中，控制器的输出与输入误差信号的微分（即误差的变化率）成正比关系。微分控制反映误差的变化率，只有当误差随时间变化时，微分控制才会对系统起作用，而对无变化或缓慢变化的对象不起作用。因此微分控制在任何情况下不能单独与被控对象串联使用，而只能构成 PD 或 PID 控制。

自动控制系统在克服误差的调节过程中可能会出现振荡甚至不稳定。原因是由于存在有较大惯性的组件（环节）或有滞后的组件，具有抑制误差的作用，其变化总是落后于误差的变化。解决的办法是使抑制误差作用的变化“超前”，即在误差接近零时，抑制误差的作用就应该是零。这就是说，在控制器中仅引入“比例”项是不够的，比例项的作用仅是放大误差的幅值，而目前需要增加的是“微分项”，它能预测误差变化的趋势，这样，

具有“比例＋微分”的控制器，就能提前使抑制误差的控制作用等于零，甚至为负值，从而避免被控量的严重超调。因此对有较大惯性或滞后的被控对象，“比例＋微分”（PD）控制器能改善系统调节过程中的动态特性。

另外，微分控制对纯滞后环节不能起到改善控制品质的作用且具有放大高频噪声信号的缺点。

在实际应用中，当设定值有突变时，为了防止由于微分控制输出的突跳，常将微分控制环节设置在反馈回路中，这种做法称为微分先行，即微分运算只对测量信号进行，而不对设定信号进行。

【例 8-2】 控制系统如图 8.7 所示，其中 $G_o(s)$ 为三阶对象：

$$G_o(s)=\frac{1}{(s+1)(s+2)(s+5)}$$

$H(s)$ 为单位反馈，采用比例微分控制，比例系数 $K_p=2$，微分系数分别取 $\tau=0,0.3,0.7,1.5,3$，试求各比例微分系数下系统的单位阶跃响应，并绘制响应曲线。

解：MATLAB 程序代码如下。

```
G=tf(1, conv( conv([1, 1], [2, 1]), [5, 1]) );
kp=2
tou=[0, 0.3, 0.7, 1.5, 3]
for i=1:5
G1=tf( [kp*tou(i), kp], 1)
sys=feedback(G1*G, 1);
step(sys)
hold on
end
gtext('tou=0')
gtext('tou=0.3')
gtext('tou=0.7')
gtext('tou=1.5')
gtext('tou=3')
```

单位阶跃响应曲线如图 8.8 所示。

从图 8.8 可以看出，仅有比例控制时系统阶跃响应有相当大的超调量和较强烈的振荡，随着微分作用的加强，系统的超调量减小，稳定性提高，上升时间减小，快速性提高。

8.3.4　积分（I）控制

具有积分控制规律的控制称为积分控制，即 I 控制，I 控制的传递函数为：

$$G_c(s)=\frac{K_I}{s} \tag{8-8}$$

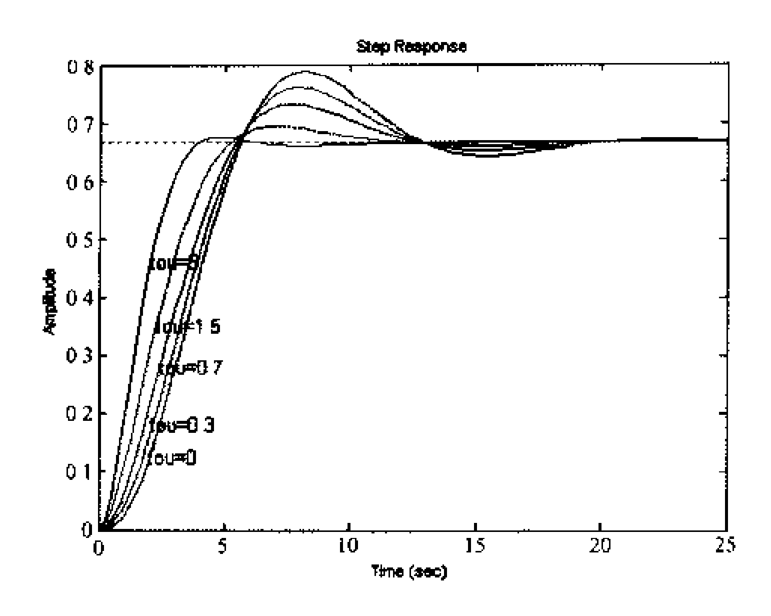


图 8.8　例 8-2 系统阶跃响应图

其中，K_I 称为积分系数。

控制器的输出信号为：

$$u(t) = K_I \int_0^t e(t)\mathrm{d}t \tag{8-9}$$

或者称，积分控制器输出信号 $u(t)$ 的变化速率与输入信号 $e(t)$ 成正比，即

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} = K_I e(t) \tag{8-10}$$

对于一个自动控制系统，如果在进入稳态后存在稳态误差，则称这个控制系统是有稳态误差的或简称有差系统。为了消除稳态误差，在控制器中必须引入积分项。积分项对误差取决于时间的积分，随着时间的增加，积分项会增大。这样，即使误差很小，积分项也会随着时间的增加而加大，它推动控制器的输出增大，使稳态误差进一步减小，直到等于零。

通常，采用积分控制的主要目的就是使系统无稳态误差，由于积分引入了相位滞后，所以使系统稳定性变差。增加积分控制对系统而言是加入了极点，对系统的响应而言是可消除稳态误差，但这对瞬时响应会造成不良影响，甚至造成不稳定，因此，积分控制一般不单独使用，通常结合比例控制器构成比例积分（PI）控制器。

8.3.5　比例积分（PI）控制

具有比例加积分控制规律的控制称为比例积分控制，即 PI 控制，PI 控制的传递函数为：

$$G_c(s) = K_P + \frac{K_P}{T_i} \cdot \frac{1}{s} = \frac{K_P\left(s + \frac{1}{T_i}\right)}{s} \tag{8-11}$$

其中，K_P 为比例系数，T_i 称为积分时间常数，两者都是可调的参数。

控制器的输出信号为：

$$u(t)=K_P e(t)+\frac{K_P}{T_i}\int_0^t e(t)\mathrm{d}t \tag{8-12}$$

PI 控制器可以使系统在进入稳态后无稳态误差。

PI 控制器与被控对象串联连接时，相当于在系统中增加了一个位于原点的开环极点，同时也增加了一个位于 S 左半平面的开环零点。位于原点的极点可以提高系统的型别，以消除或减小系统的稳态误差，改善系统的稳态性能；而增加的负实部零点则可减小系统的阻尼比，缓和 PI 控制器极点对系统稳定性及动态过程产生的不利影响。在实际工程中，PI 控制器通常用来改善系统的稳态性能。

【例 8-3】 单位负反馈控制系统的开环传递函数 $G_o(s)$ 为：

$$G_o(s)=\frac{1}{(s+1)(2s+1)(5s+1)}$$

采用比例积分控制，比例系数 $K_P=2$，积分时间常数分别取 $T_i=3,6,14,21,28$，试求各比例积分系数下系统的单位阶跃响应，并绘制响应曲线。

解：MATLAB 程序代码如下。

```
G=tf(1, conv( conv([1, 1], [2, 1]), [5, 1]) );
kp=2
ti=[3, 6, 14, 21, 28]
for i=1:5
G1=tf( [kp, kp/ti(i)], [1, 0])
sys=feedback(G1*G, 1);
step(sys)
hold on
end
gtext('ti=3')
gtext('ti=6')
gtext('ti=14')
gtext('ti=21')
gtext('ti=28')
```

响应曲线如图 8.9 所示。

从图 8.9 中可以看出，随着积分时间的减小，积分控制作用增强，闭环系统的稳定性变差。

8.3.6 比例积分微分（PID）控制

具有比例加积分加微分控制规律的控制称为比例积分微分控制，即 PID 控制，PID 控制的传递函数为：

$$G_c(s)=K_P+\frac{K_P}{T_i}\cdot\frac{1}{s}+K_P\tau s \tag{8-13}$$

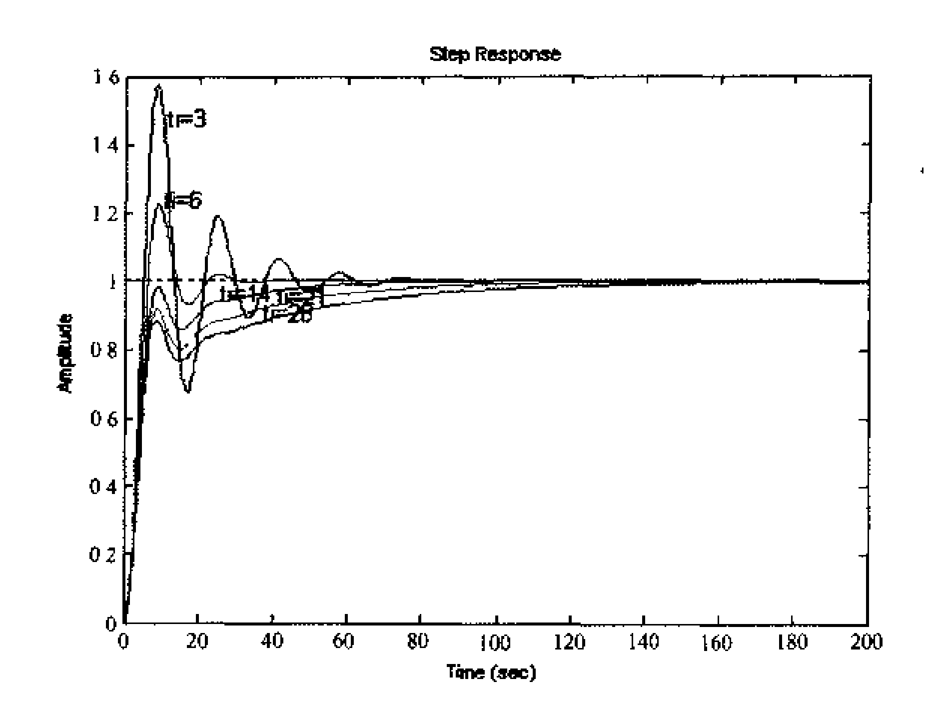


图 8.9　例 8-3 系统阶跃响应图

式中，K_P 为比例系数，T_i 称为积分时间常数，τ 为微分时间常数，三者都是可调的参数。

PID 控制器的输出信号为：

$$u(t)=K_P e(t)+\frac{K_P}{T_i}\int_0^t e(t)\mathrm{d}t+K_P\tau\frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{8-14}$$

PID 控制器的传递函数可写成：

$$\frac{U(s)}{E(s)}=\frac{K_P}{T_i}\cdot\frac{(T_i\tau s^2+T_i s+1)}{s} \tag{8-15}$$

PI 控制器与被控对象串联连接时，可以使系统的型别提高一级，而且还提供了两个负实部的零点。与 PI 控制器相比，PID 控制器除了同样具有提高系统稳态性能的优点外，还多提供了一个负实部零点，因此在提高系统动态性能方面具有更大的优越性。在实际工程中，PID 控制器被广泛应用。

PID 控制通过积分作用消除误差，而微分控制可缩小超越量、加快系统响应，是综合了 PI 控制与 PD 控制长处并去除其短处的控制。从频域角度来看，PID 控制是通过积分作用于系统的低频段，以提高系统的稳态性能；而微分作用于系统的中频段，以改善系统的动态性能。

8.3.7　PID 控制器参数整定

PID 控制器的参数整定是控制系统设计的核心内容，它根据被控过程的特性确定 PID 控制器的比例系数、积分时间和微分时间。

PID 控制器参数整定的方法很多，概括起来有两大类：

（1）理论计算整定法，主要依据系统的数学模型，经过理论计算确定控制器参数。这种方法所得到的计算数据未必可以直接使用，还必须通过工程实际进行调整和修改。

（2）工程整定方法，主要有 Ziegler—Nichols 整定法、临界比例度法、衰减曲线法。这三种方法各有特点，其共同点都是通过试验然后按照工程经验公式对控制器参数进行整定。无论采用哪一种方法所得到的控制器参数，都需要在实际运行中进行最后调整与完善。工程整定法的基本特点是：不需要事先知道过程的数学模型，直接在过程控制系统中进行现场整定，方法简单，计算简便，易于掌握。

8.3.7.1 Ziegler—Nichols 整定方法

Ziegler—Nichols 法是一种基于频域设计 PID 控制器的方法。基于频域的参数整定是需要参考模型的，首先需要辨识出一个能较好反映被控对象频域特性的二阶模型。根据这样的模型，结合给定的性能指标可推导出公式，以用于 PID 参数的整定。

基于频域的设计方法在一定程度上回避了精确的系统建模，而且有较明确的物理意义，比常规的 PID 控制有更多的可适应场合。目前已经有一些基于频域设计 PID 控制器的方法，如 Ziegler—Nichols 法、Cohen—Coon 法等。Ziegler—Nichols 法是最常用的整定 PID 参数的方法。

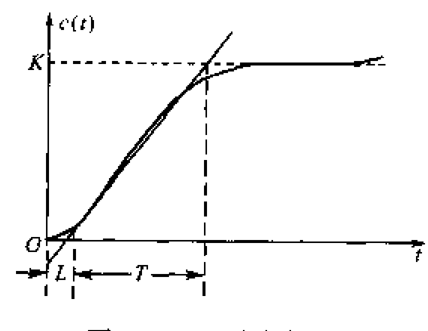


图 8.10　S 形响应曲线

Ziegler—Nichols 法是根据给定对象的瞬态响应特性来确定 PID 控制器的参数。Ziegler—Nichols 法首先通过实验获取控制对象单位阶跃响应，如图 8.10 所示。

如果单位阶跃响应曲线看起来是一条 S 形的曲线，则可用此法，否则不能用。S 形曲线用延迟时间 L 和时间常数 T 来描述，则对象传递函数可近似为：

$$\frac{C(s)}{R(s)}=\frac{K\mathrm{e}^{-Ls}}{Ts+1} \tag{8-16}$$

利用延迟时间 L、放大系数 K 和时间常数 T，根据表 8.1 中的公式确定 K_p、T_i 和 τ 的值。

表 8.1　Ziegler—Nichols 法整定控制器参数

控制器类型	比例度 δ / %	积分时间 T_i	微分时间 τ
P	$\frac{T}{(K\cdot L)}$	∞	0
PI	$0.9\frac{T}{(K\cdot L)}$	$\frac{L}{0.3}$	0
PID	$1.2\frac{T}{(K\cdot L)}$	$2.2L$	$0.5L$

【例 8-4】　已知如图 8.11 所示的控制系统。

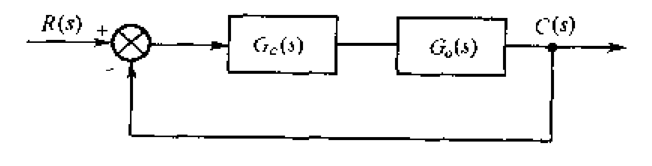


图 8.11　控制系统结构图

系统开环传递函数$G_o(s)$为：

$$G_o(s)=\frac{8}{(360s+1)}e^{-180s}$$

试采用 Ziegler—Nichols 整定公式计算系统 P、PI、PID 控制器的参数，并绘制整定后系统的单位阶跃响应曲线。

解：PID 参数整定是一个反复调整测试的过程，使用 Simulink 能大大简化这一过程。根据题意，建立如图 8.12 所示的 Simulink 模型。

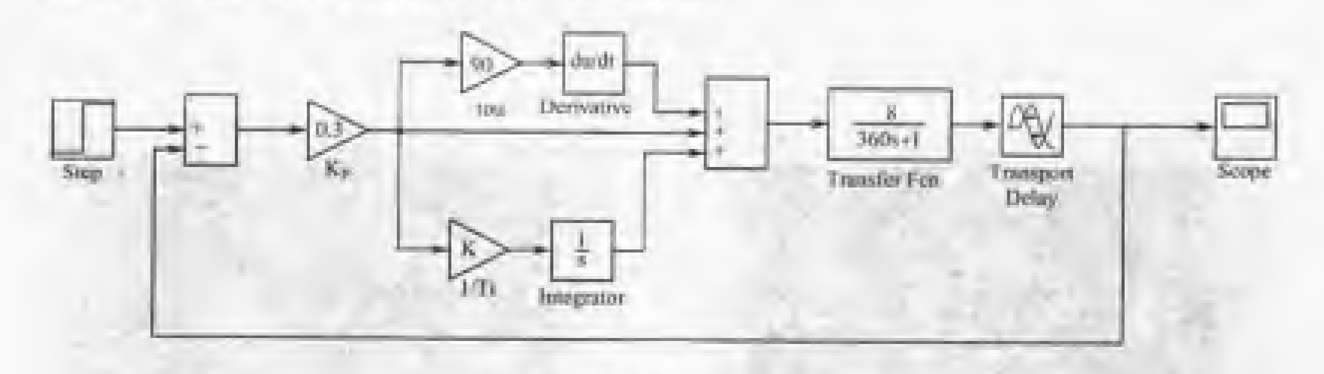


图 8.12　例 8-4 系统 Simulink 模型

图中，“Integrator”为积分器，“Derivative”为微分器，“Kp”为比例系数K_p，“1/Ti”为积分时间常数T_i，“tou”为微分时间常数τ。进行 P 控制器参数整定时，微分器和积分器的输出不连到系统中，在 Simulink 中，把微分器和积分器的输出连线断开即可。同理，进行 PI 控制器参数整定时，微分器的输出连线断开。

Ziegler—Nichols 整定的第一步是获取开环系统的单位阶跃响应，在 Simulink 中，把反馈连线、微分器的输出连线、积分器的输出连线都断开，“Kp”的值置为 1，选定仿真时间（注意，如果系统滞后比较大，则应相应加大仿真时间），仿真运行，运行完毕后，双击“Scope”，得到如图 8.13 所示的结果。



图 8.13　例 8-4 系统开环单位阶跃响应曲线

按照 S 形响应曲线的参数求法，大致可以得到系统延迟时间L、放大系数K和时间常数T如下：

$$L = 180,\quad T = 110 - 80 = 360,\quad K = 8$$

如果从示波器的输出不易看出这 3 个参数，则可以将系统输出导入到 MATLAB 的工作空间中，然后编写相应的 m 文件求取这 3 个参数。

根据表 8.1，可知 P 控制整定时，比例放大系数 $K_p = 0.25$，将“Kp”的值置为 0.25，仿真运行，运行完毕后，双击“Scope”得到如图 8.14 所示的结果，它是 P 控制时系统的单位阶跃响应。

根据表 8.1，可知 PI 控制整定时，比例放大系数 $K_p = 0.225$，积分时间常数 $T_i = 594$，将“Kp”的值置为 0.225，“1/Ti”的值置为 1/594，将积分器的输出连线连上，仿真运行，运行完毕后，双击“Scope”得到如图 8.15 所示的结果，它是 PI 控制时系统的单位阶跃响应。

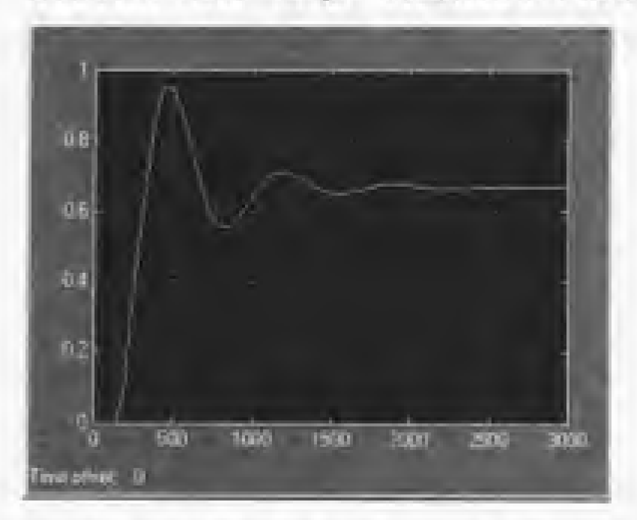


图 8.14　例 8-4 系统 P 控制时的单位阶跃响应曲线

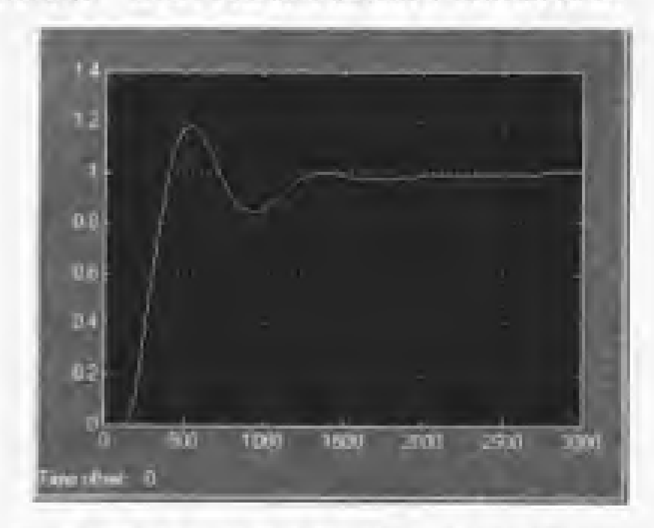


图 8.15　例 8-4 系统 PI 控制时的单位阶跃响应曲线

根据表 8.1，可知 PID 控制整定时，比例放大系数 $K_p = 0.3$，积分时间常数 $T_i = 396$，微分时间常数 $\tau = 90$，将“Kp”的值置为 0.3，“1/Ti”的值置为 1/396，“tou”的值置为 90，将微分器的输出连线连上，仿真运行，运行完毕后，双击“Scope”得到如图 8.16 所示的结果，它是 PID 控制时系统的单位阶跃响应。

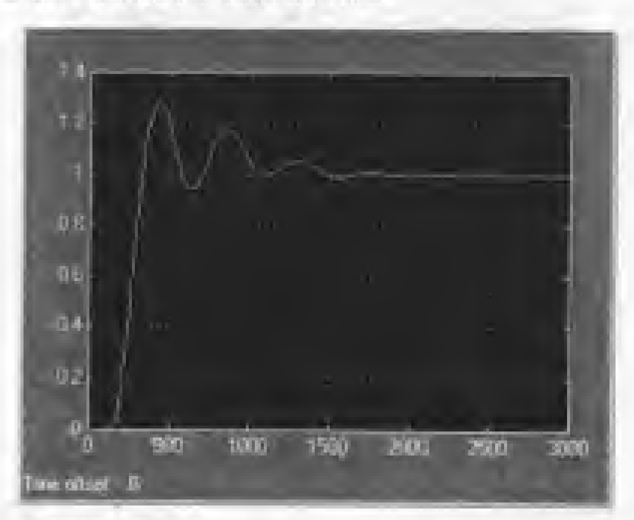


图 8.16　例 8-4 系统 PID 控制时的单位阶跃响应曲线

由图 8.14，图 8.15，图 8.16 对比可以看出，P 控制和 PI 控制两者的响应速度基本相同，因为这两种控制的比例系数不同，因此系统稳定的输出值不同。PI 控制的超调量比 P 控制的要小，PID 控制比 P 控制和 PI 控制的响应速度要快，但是超调量大些。

【例 8-5】 已知如图 8.11 所示的控制系统，其中系统开环传递函数 $G_o(s)$ 为：

$$G_o(s)=\frac{1.67}{(4.05s+1)}\cdot\frac{8.22}{(s+1)}e^{-1.5s}$$

试采用 Ziegler—Nichols 整定公式计算系统 P、PI、PID 控制器的参数，并绘制整定后系统的单位阶跃响应曲线。

解：根据题意，建立如图 8.17 所示的 Simulink 模型。

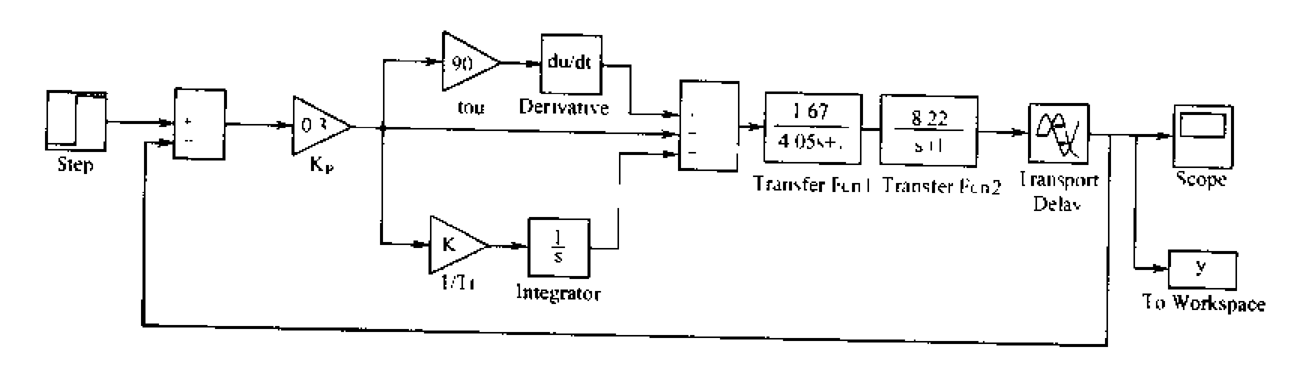


图 8.17　例 8-5 系统 Simulink 模型

Ziegler—Nichols 整定的第一步是获取开环系统的单位阶跃响应，在 Simulink 中，把反馈连线、微分器的输出连线、积分器的输出连线都断开，"Kp" 的值置为 1，选定仿真时间（注意，如果系统滞后比较大，则应相应加大仿真时间），仿真运行，运行完毕后，双击 "**Scope**" 得到如图 8.18 所示的结果。

按照 S 形响应曲线的参数求法，大致可以得到系统延迟时间 L、放大系数 K 和时间常数 T 如下：

$$L=2.2\text{，}T=9.2-2.2=7\text{，}K=13.727$$

如果从示波器的输出不好看出这 3 个参数，可以将系统输出导入到 MATLAB 的工作空间中，然后编写相应的 m 文件求取这 3 个参数。

根据表 8.1，可知 P 控制整定时，比例放大系数 $K_p=0.2318$，将"Kp"的值置为 0.2318，仿真运行，运行完毕后，双击 "Scope" 得到如图 8.19 所示的结果，它是 P 控制时系统的单位阶跃响应。

根据表8.1，可知 PI 控制整定时，比例放大系数 $K_p=0.2086$，积分时间常数 $T_i=7.3333$，将 "Kp" 的值置为 0.2086，"1/T₁" 的值置为 1/7.3333，将积分器的输出连线连上，仿真运行，运行完毕后，双击 "Scope" 得到如图 8.20 所示的结果，它是 PI 控制时系统的单位阶跃响应。

根据表 8.1，可知 PID 控制整定时，比例放大系数 $K_p=0.3$，积分时间常数 $T_i=4.84$，微分时间常数 $\tau=1.1$，将 "Kp" 的值置为 0.3，"1/T₁" 的值置为 1/4.84，"tou" 的值置为 1.1，将微分器的输出连线连上，仿真运行，运行完毕后，双击 "Scope" 得到如图 8.21

所示的结果，它是 PID 控制时系统的单位阶跃响应。

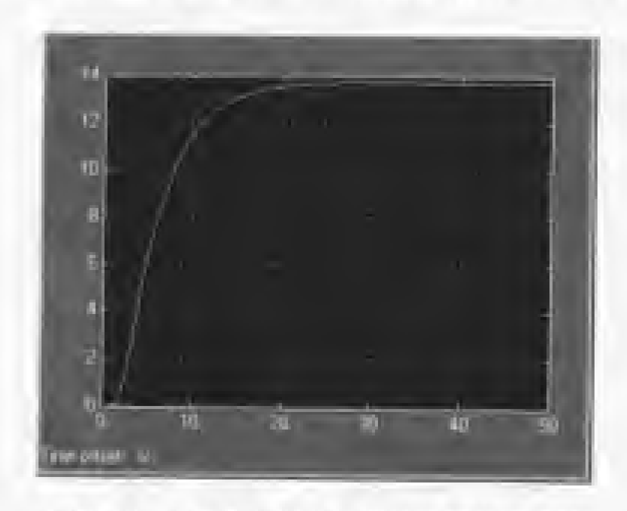


图 8.18　例 8-5 系统开环单位阶跃响应曲线



图 8.19　例 8-5 系统 P 控制时的单位阶跃响应曲线

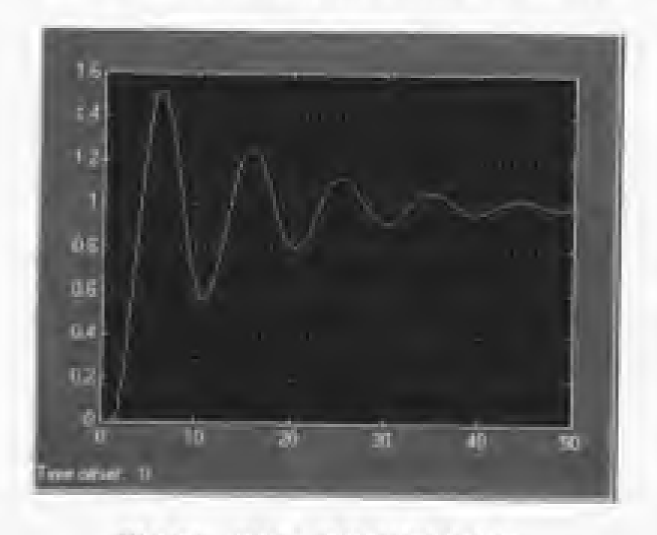


图 8.20　例 8-5 系统 PI 控制时的单位阶跃响应曲线

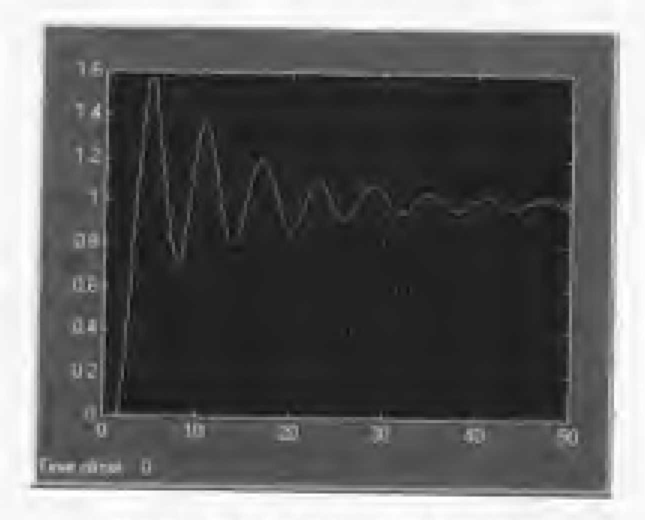


图 8.21　例 8-5 系统 PID 控制时的单位阶跃响应曲线

由图 8.19，图 8.20 和图 8.21 对比可以看出，P 控制和 PI 控制两者的响应速度基本相同，超调量大不相同，但由于这两种控制的比例系数不同，因此系统稳定的输出值不同。PI 控制的超调量比 P 控制的要小，PID 控制比 P 控制和 PI 控制的响应速度要快，但是超调量大些。

8.3.7.2　临界比例度法

临界比例度法适用于已知对象传递函数的场合，在闭合的控制系统里，将调节器置于纯比例作用下，从大到小逐渐改变调节器的比例度，得到等幅振荡的过渡过程。此时的比例度称为临界比例度 δ_k，相邻两个波峰间的时间间隔称为临界振荡周期 T_k。采用临界比例度法时，系统产生临界振荡的条件是系统的阶数是 3 阶或 3 阶以上。

临界比例度法的步骤如下：

（1）将调节器的积分时间 T_i 置于最大（$T_i=\infty$），微分时间置零（$\tau=0$），比例度 δ 适当，平衡操作一段时间，把系统投入自动运行。

（2）将比例度 δ 逐渐减小，得到等幅振荡过程，记下临界比例度 δ_k 和临界振荡周期 T_k 的值。

（3）根据 δ_k 和 T_k 的值，采用表 8.2 中的经验公式，计算出调节器的各个参数，即 δ、T_i 和 τ 的值。

表 8.2 临界比例度法整定控制器参数

控制器类型	比例度 δ/%	积分时间 T_i	微分时间 τ
P	$2\delta_k$	∞	0
PI	$2.2\delta_k$	$0.833T_k$	0
PID	$1.7\delta_k$	$0.50T_k$	$0.125T_k$

按"先 P 后 I 最后 D"的操作程序将调节器整定参数调到计算值上。若还不够满意，可再进一步调整。

临界比例度法整定的注意事项：

- 有的过程控制系统，临界比例度很小，调节阀不是全关就是全开，对工业生产不利。
- 有的过程控制系统，当调节器比例度 δ 调到最小刻度值时，系统仍不产生等幅振荡，对此，将最小刻度的比例度作为临界比例度 δ_k 进行调节器参数整定。

【例 8-6】 已知如图 8.11 所示的控制系统，其中系统开环传递函数 $G_o(s)$ 为：

$$G_o(s)=\frac{1}{s(s+1)(s+5)}$$

试采用临界比例度法计算系统 P、PI、PID 控制器的参数，并绘制整定后系统的单位阶跃响应曲线。

解：根据题意，建立如图 8.22 所示的 Simulink 模型。

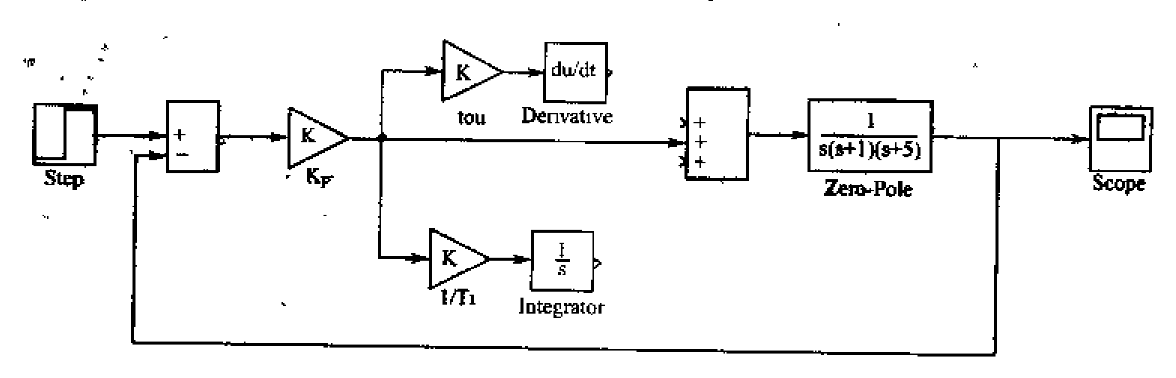


图 8.22 例 8-6 系统 Simulink 模型

临界比例度法整定的第一步是获取系统的等幅振荡曲线，在 Simulink 中，把反馈连线、微分器的输出连线、积分器的输出连线都断开，"Kp"的值从大到小进行试验，每次仿真结束后，观察示波器的输出，直到输出等幅振荡曲线为止。本例中当 $K_p=30$ 时出现等幅振荡，此时的 $T_k=2.81$，等幅振荡曲线如图 8.23 所示。

根据表 8.2，可知 P 控制整定时，比例放大系数 $K_p=15$，将 "Kp" 的值置为 15，仿真运行，运行完毕后，双击 "Scope" 得到如图 8.24 所示的结果，它是 P 控制时系统的单位阶跃响应。

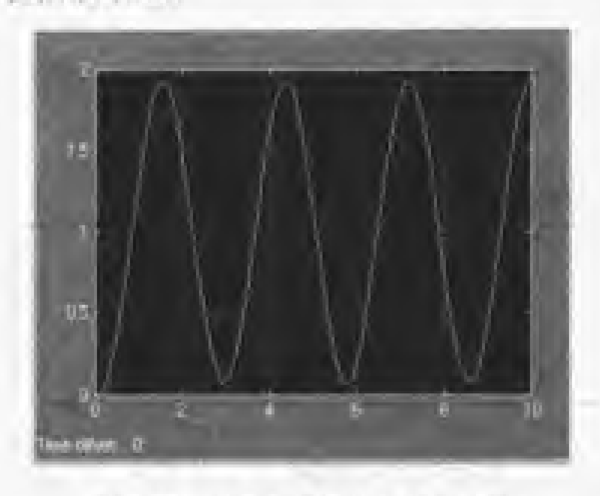


图 8.23　例 8-6 系统等幅振荡曲线

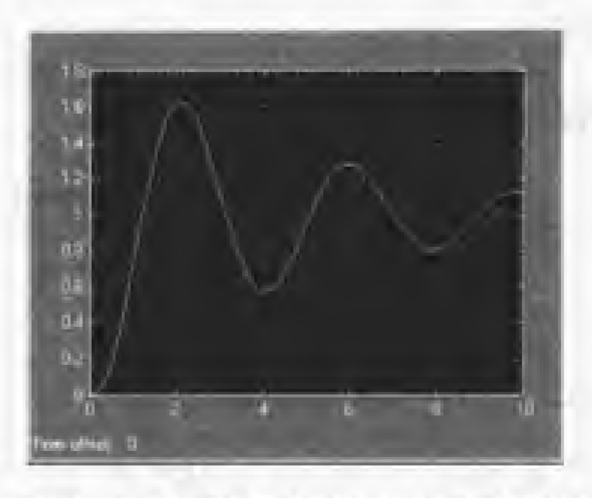


图 8.24　例 8-6 系统 P 控制时的单位阶跃响应曲线

根据表 8.2，可知 PI 控制整定时，比例放大系数 $K_p=13.5$，积分时间常数 $T_i=2.3417$，将 "Kp" 的值置为 13.5，"1/Ti" 的值置为 1/2.3417，将积分器的输出连线连上，仿真运行，运行完毕后，双击 "Scope" 得到如图 8.25 所示的结果，它是 PI 控制时系统的单位阶跃响应。

根据表 8.2，可知 PID 控制整定时，比例放大系数 $K_p=17.6471$，积分时间常数 $T_i=1.405$，微分时间常数 $\tau=0.35124$，将"Kp"的值置为 17.6471，"1/Ti"的值置为 1/1.405，"tou"的值置为 0.35124，将微分器的输出连线连上，仿真运行，运行完毕后，双击"Scope"得到如图 8.26 所示的结果，它是 PID 控制时系统的单位阶跃响应。

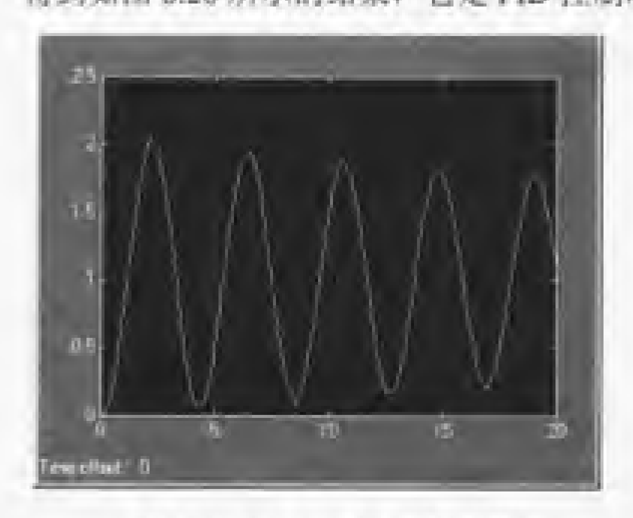


图 8.25　例 8-6 系统 PI 控制时的单位阶跃响应曲线

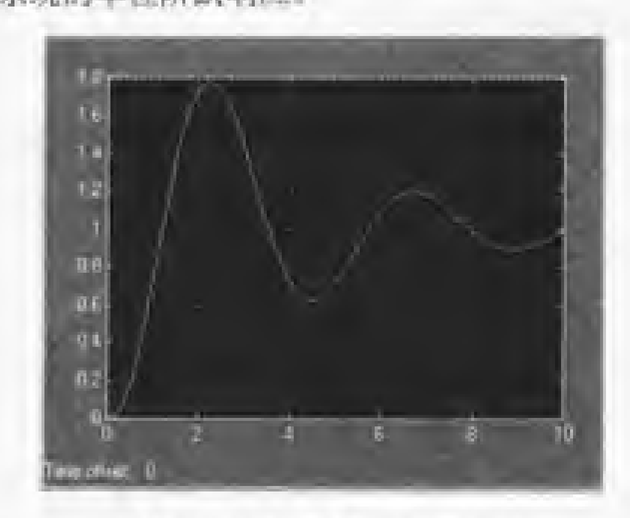


图 8.26　例 8-6 系统 PID 控制时的单位阶跃响应曲线

由图 8.24，图 8.25 和图 8.26 对比可以看出，P 控制和 PI 控制的阶跃响应上升速度基本相同，由于这两种控制的比例系数不同，因此系统稳定的输出值不同。PI 控制的超调

量比 P 控制的要小，PID 控制比 P 控制和 PI 控制的响应速度要快，但是超调量大些。

值得注意的是，由于工程整定方法依据的是经验公式，不是在任何情况下都适用的，因此，按照经验公式整定的 PID 参数并不是最好的，需要进行一些调整。本例中，按照表 8.2 整定的 PI 控制器的参数就不是非常好，这从图 8.25 中可以看出。将比例放大系数调整为 $K_p = 13.5$，积分时间常数调整为 $T_i = 12.5$，仿真运行，运行完毕后，双击“Scope”得到如图 8.27 所示的结果。

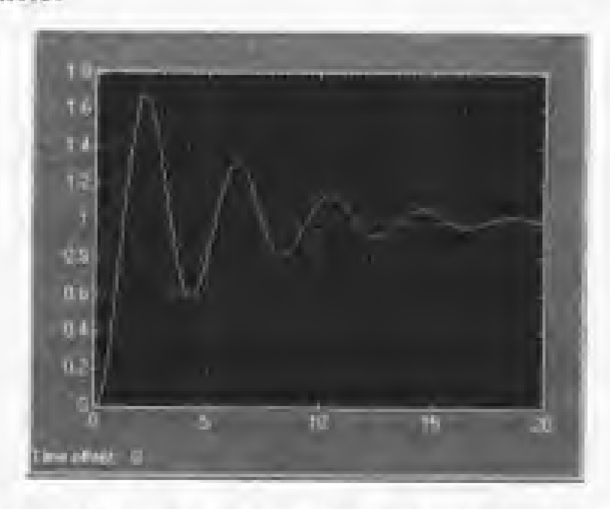


图 8.27　例 8-6 系统调整 PI 参数后的单位阶跃响应曲线

对比图 8.27 和图 8.25 可以看出，调整 PI 参数后系统的超调量减小了，调节时间也减小了。当然，调整参数的方法有多种，既可以调整 P 的参数，也可以调整 I 的参数，也可以同时调整这两者的参数。

8.3.7.3　衰减曲线法

衰减曲线法根据衰减频率特性整定控制器参数。先把控制系统中调节器参数置成纯比例作用（$T_i = \infty$，$\tau = 0$），使系统投入运行，再把比例度 δ 从大逐渐调小，直到出现 4:1 衰减过程曲线，如图 8.28 所示。

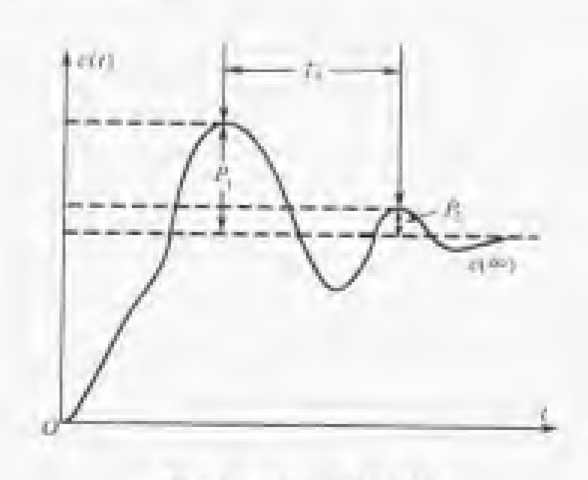


图 8.28　4:1 衰减曲线

此时的比例度为 4:1，即 $\frac{P_1}{P_2}=4:1$，衰减比例度为 δ_s，上升时间为 t_r，两个相邻波峰间的时间间隔 T_s 称为 4:1 衰减振荡周期。

根据 δ_s、t_r 或 T_s，使用表 8.3 所示的经验公式，即可计算出调节器的各个整定参数值。

表 8.3　衰减曲线法整定控制器参数

控制器类型	比例度 δ/%	积分时间 T_I	微分时间 τ
P	δ_s	∞	0
PI	$1.2\delta_s$	$2t_r$ 或 $0.5T_s$	0
PID	$0.8\delta_s$	$1.2t_r$ 或 $0.3T_s$	$0.4t_r$ 或 $0.17T_s$

按“先 P 后 I 最后 D”的操作顺序，将求得的整定参数设置在调节器上，再观察运行曲线，若不太理想，还可适当调整。

衰减曲线法的注意事项：

（1）反应较快的控制系统，要认定 4:1 衰减曲线和读出 T_s 比较困难，此时，可用记录指针来回摆动两次就达到稳定作为 4:1 衰减过程。

（2）在生产过程中，负荷变化会影响过程特性。当负荷变化较大时，必须重新整定调节器参数值。

（3）若认为 4:1 衰减太慢，可采用 10:1 衰减过程。对于 10:1 衰减曲线法整定调节器参数的步骤与上述完全相同，仅所用计算公式有些不同，具体公式可查阅相关资料，此处不再赘述。

【例 8-7】　已知如图 8.11 所示的控制系统，其中系统开环传递函数 $G_o(s)$ 为：

$$G_o(s)=\frac{6}{(s+1)(s+2)(s+3)}$$

试采用临界比例度法计算系统 P、PI、PID 控制器的参数，并绘制整定后系统的单位阶跃响应曲线。

解：根据题意，建立如图 8.29 所示的 Simulink 模型。

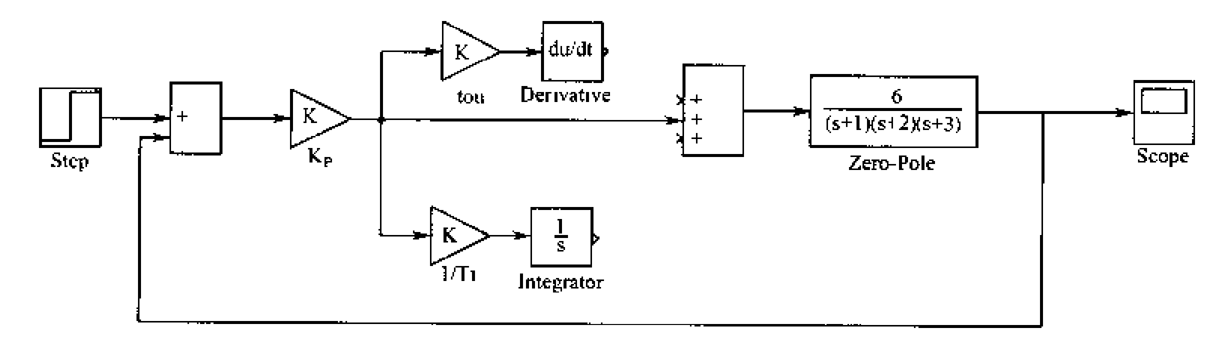


图 8.29　例 8-7 系统 Simulink 模型

衰减曲线法整定的第一步是获取系统的衰减曲线，本例按 4:1 衰减曲线整定，在 Simulink 中，把反馈连线、微分器的输出连线、积分器的输出连线都断开，“K_P”的值从大到小进行试验，每次仿真结束后，观察示波器的输出，直到输出 4:1 衰减振荡曲线为

止，当 $K_p = 3.823$ 时，在 $t=1.55$ 时出现第一峰值，它的值为 1.13；在 $t=4.24$ 时出现第二峰值，它的值为 0.88，稳定值是 0.8，计算可得衰减度为 4:1。因此，当 $K_p = 3.823$ 时，系统出现 4:1 衰减振荡，且 $T_s = 4.24 - 1.55 = 2.69$，曲线如图 8.30 所示。

根据表 8.3，可知 P 控制整定时，比例放大系数和出现 4:1 衰减振荡时的比例系数相同，因此，P 控制时系统的单位阶跃响应曲线和图 8.30 相同。

根据表 8.3，可知 PI 控制整定时，比例放大系数 $K_p = 3.1858$，积分时间常数 $T_i = 1.345$，将“Kp”的值置为 3.1815，“1/Ti”的值置为 1/1.345，将积分器的输出连线连上，仿真运行，运行完毕后，双击“Scope”得到如图 8.31 所示的结果，它是 PI 控制时系统的单位阶跃响应曲线。

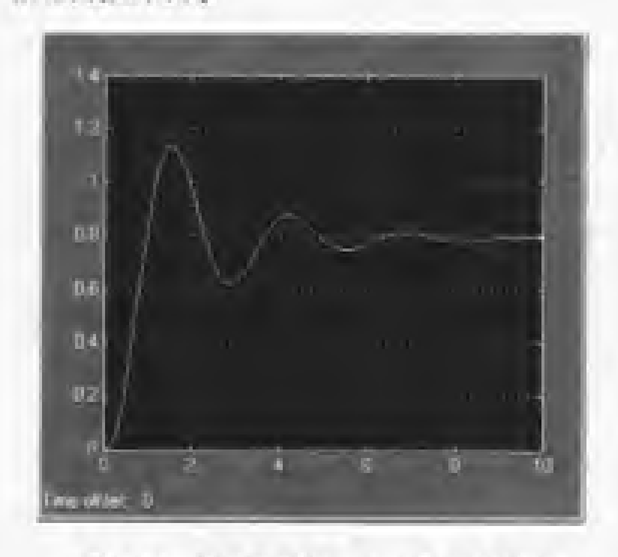


图 8.30　例 8-7 系统 4:1 衰减振荡曲线

图 8.31　例 8-7 系统 PI 控制时的单位阶跃响应曲线

根据表 8.3，可知 PID 控制整定时，比例放大系数 $K_p = 4.7787$，积分时间常数 $T_i = 0.807$，微分时间常数 $\tau = 0.269$，将“Kp”的值置为 4.7787，“1/Ti”的值置为 1/0.807，“tou”的值置为 0.269，将微分器的输出连线连上，仿真运行，运行完毕后，双击“Scope”得到如图 8.32 所示的结果，它是 PID 控制时系统的单位阶跃响应曲线。

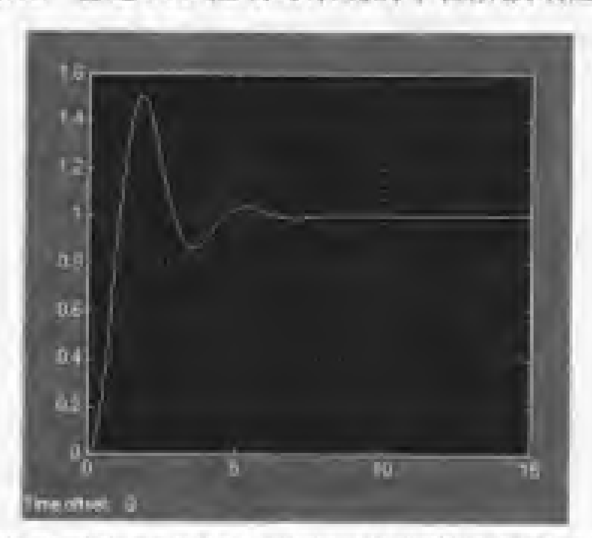


图 8.32　例 8-7 系统 PID 控制时的单位阶跃响应曲线

由图 8.30，图 8.31 和图 8.32 对比可以看出，P 控制和 PI 控制的阶跃响应上升速度基本相同，由于这两种控制的比例系数不同，因此系统稳定的输出值不同。PI 控制的超调量比 P 控制的要小，PID 控制比 P 控制和 PI 控制的响应速度要快，但是超调量大些。

在 PID 参数进行整定时，如果能够有理论的方法确定 PID 参数当然是最理想的方法，但是在实际应用中，更多的是通过凑试法来确定 PID 的参数。通过上面的例子，可以总结出几条基本的 PID 参数整定规律：

（1）增大比例系数一般将加快系统的响应，在有静差的情况下有利于减小静差，但是过大的比例系数会使系统有比较大的超调，并产生振荡，使稳定性变差。

（2）增大积分时间有利于减小超调、减小振荡，使系统的稳定性增加，但是系统静差消除时间变长。

（3）增大微分时间有利于加快系统的响应速度，使系统超调量减小，稳定性增加，但系统对扰动的抑制能为减弱。

在凑试时，可参考以上参数对系统控制过程的影响超势，对参数调整实行先比例、后积分、再微分的整定步骤。即先整定比例部分，将比例参数由小变大，并观察相应的系统响应，直至得到反应快、超调小的响应曲线。如果系统没有静差或得超已超小到允许范围内，并且对响应曲快已超满意，则只需要比例调节得即可。

如果在比例调节的基础上系统的静差不能满足设计变求，则必须加入积分环节。在超定时先将积分时间设定到一个比较大的值，然后将已超调节好的比例系数略为缩小（一般缩小为原值的 0.8 倍），然后减小积分时间，使得系统在保得良好动态性能的快况下，静差得到消除。在此过程中，可根据系统的响应曲线的好坏反复改变比例系数和积分时间，以期得到将意的控制过程和整定参数。

如果在上述调整过程中对系统的动态过程反复调整还不能得到满意的结果，则可以加入微分环节。首先把值分时间设置为 0，在上述基础上逐渐增加微分时间，同时相应性改变比例系数和积分时间，逐步凑试，直至得到满意的调节效果。

8.4　控制系统校正的根轨迹法

根轨迹法是一种图解法，它值动了系统某一参数（通常是增益）从零变化到无穷大时其闭环极点位置的变化。但在实际中，只调整增益通常是不能获得所希望的性能的，因此，必须改造根轨迹，通过引入超当的校正装置来改变原来的根轨迹。引入校正装置就是在系统中增加零点和（或）极点，通过零极点的变化改变根轨迹的形状。

用根轨迹法进行校正的基扰，是通过在系统开环传递函数中增加零点和极点以改变根轨迹的形状，从而使系统值轨迹在 S 平面上通过希望缩闭环极点。校轨迹法校正的特征是基于闭环系统具有一对主导闭环极点，当然，零点和附加的极点会影响响应特性。

应用根轨迹进行快正，实质上是通过采用校正装置改变根轨迹，从而将一对主导闭环极点配置到期望的位置上。

在开环传递函数中增加极点，可以值根轨迹向右方得动，从而降低系统的相对稳定性，增大系统调节时间。前面讲的积分控制，相当于给系统增加了位于原点的极点，因此降低了系统的稳定性。

在开环传递函数中增加零点，可以使根轨迹向左方移动，从而提高系统的相对稳定性，减小系统调节时间。前面讲的微分控制，相当于给系统前向通道中增加了零点，因此增加了系统的超调量，并且加快了瞬态响应。

当系统的性能指标是以最大超调量、上升时间、调整时间、阻尼比以及希望的闭环阻尼比、闭环极点无阻尼振频率等表示时，采用根轨迹法进行校正比较方便，在设计系统时，如果需要对增益以外的参数进行调整，则必须通过引入适当的校正装置来改变原来的零极点。

采用根轨迹法确定串联校正参数的条件是：

- 已确定采用串联校正方案；
- 给定时域指标 $\sigma_p, t_s, e_{ss}(\infty)$ 。

设已知系统不可变部分的传递函数为：

$$G_o(s) = k\frac{(s-z_1)(s-z_2)\cdots(s-z_m)}{s^{v}(s-p_1)(s-p_2)\cdots(s-p_n)} \tag{8-17}$$

式中， K 为开环增益， $K = \lim_{s\to 0} s^{v} G_o(s) = k\dfrac{\prod_{i=1}^{m}(-z_i)}{\prod_{i=1}^{n-v}(-p_i)}$ ，开环极点 $p_i(i=1,2,3,\cdots,n-v)$ 和零点 $z_i(i=1,2,3,\cdots,m)$ 为已知数据。

8.4.1 基于根轨迹法的超前校正

用根轨迹法设计超前校正装置的步骤为：

（1）先假定系统的控制性能由靠虚轴最近的一对闭环共轭极点 s_d 来主导。

（2）应用二阶系统参量 ζ 和 ω_n 与时域指标间的关系，按给定的 σ_p 与 t_s 确定闭环主导极点的位置。.

（3）绘制原系统根轨迹，如果根轨迹不能通过希望的闭环主导极点，则表明仅调整增益不能满足给定要求，需加校正装置。如果原系统根轨迹位于期望极点的右侧，则应加入超前校正装置。

（4）计算超前校正装置应提供的超前相角：

$$\varphi_c = \pm(2k+1)\pi - \angle G_o(s_d) \tag{8-18}$$

（5）按式（8-18）求校正装置零点、极点位置。

（6）由幅值条件确定校正后系统增益。

（7）校验系统的性能指标，如果系统不能满足要求指标，适当调整零点、极点位置。如果需要大的静态误差系数，则应采用其他方案。

【例 8-8】 已知系统开环传递函数为：

$$G_o(s) = \frac{2.3}{s(1+0.2s)(1+0.15s)}$$

试设计超前校正环节，使其校正后系统的静态速度误差系数 $K_v \leqslant 4.6$ ，闭环主导极点满足阻尼比 $\zeta = 0.2$ ，自然振荡角频率 $\omega_n = 12.0\text{rad/s}$ ，并绘制校正前后系统的单位阶跃

响应曲线、单位脉冲响应曲线和根轨迹。

解：计算串联超前校正环节参数的子函数 MATLAB 程序代码如下。

```
function Gc=cqjz_root(G, s1, kc)
numG=G.num{1}
denG=G.den{1}
ngv=polyval(numG, s1)
dgv=polyval(denG, s1)
g=ngv/dgv
theta_G=angle(g)
theta_s=angle(s1)
MG=abs(g)
Ms=abs(s1)
Tz=( sin(theta_s)-kc*MG*sin(theta_G-theta_s) )/( kc*MG*Ms*sin(theta_G) )
Tp=-( kc*MG*sin(theta_s)+sin(theta_G+theta_s))/(Ms*sin(theta_G))
Gc=tf([Tz, 1], [Tp, 1])
```

主函数 MATLAB 代码如下：

```
num=2.3
den=conv([1, 0], conv([0.2, 1], [0.15, 1]) )
G=tf(num, den)
zeta=0.2
wn=12.0
[num, den]=ord2(wn, zeta)
s=roots(den)
s1=s(1)
kc=2
Gc=cqjz_root(G, s1, kc)
GGc=G*Gc*kc
Gy_close=feedback(G, 1)
Gx_close=feedback(GGc, 1)
figure(1)
step(Gx_close, 'b', 3.5)
hold on
step(Gy_close, 'r', 3.5)
grid
gtext('校正前的')
gtext('校正后的')
figure(2)
impulse(Gx_close, 'b', 3.5)
hold on
```

```
impulse(Gy_close, 'r', 3.5)
grid
gtext('校正前的')
gtext('校正后的')
figure(3)
rlocus(G, GGc)
grid
gtext('校正前的')
gtext('校正后的')
```

运行结果如下：

```
%超前校正环节传递函数
Transfer function:
1.016 s + 1
-----------
0.0404 s + 1
%原系统闭环传递函数
Transfer function:
                 2.3
----------------------------
0.03 s^3 + 0.35 s^2 + s + 2.3
%校正后系统闭环传递函数

Transfer function:
                         4.672 s + 4.6
----------------------------------------------------------
0.001212 s^4 + 0.04414 s^3 + 0.3904 s^2 + 5.672 s + 4.6
```

系统校正前后的单位阶跃响应曲线如图 8.33 所示。

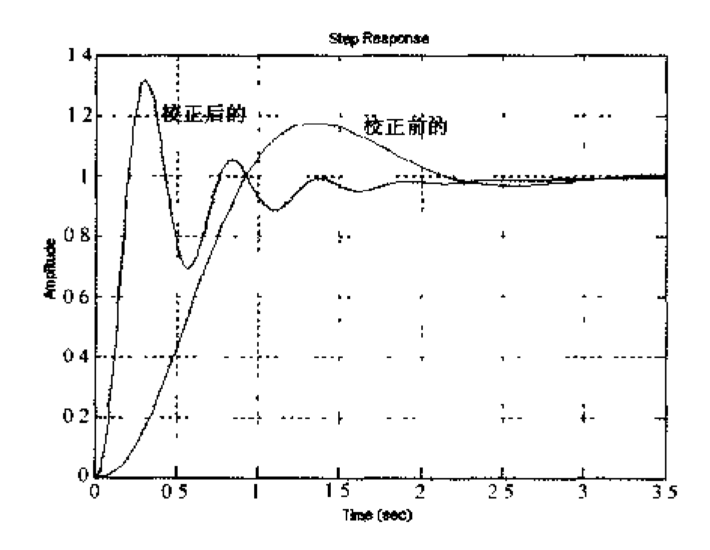


图 8.33　例 8-8 系统校正前后单位阶跃响应曲线

系统校正前后的单位脉冲响应曲线如图 8.34 所示。

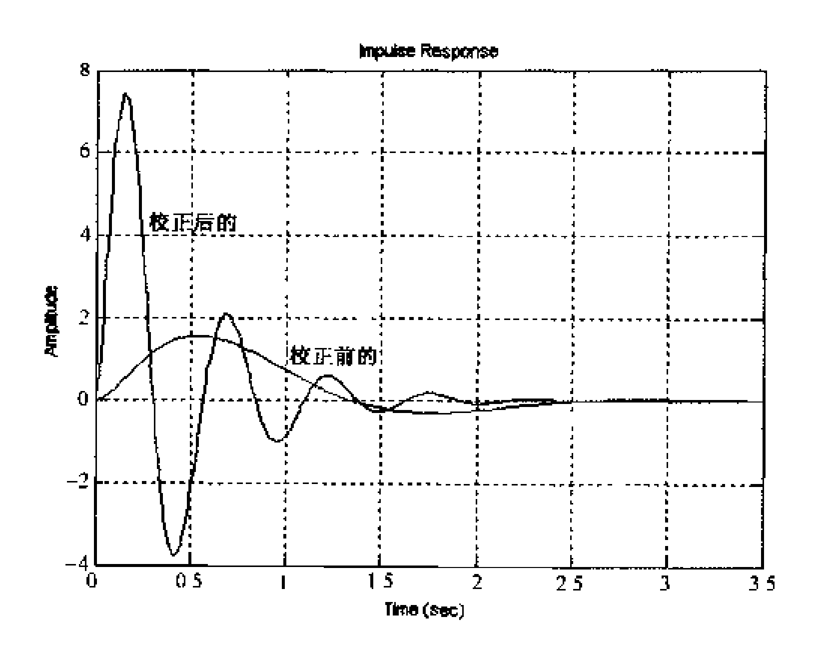


图 8.34　例 8-8 系统校正前后单位脉冲响应曲线

系统校正前后的根轨迹曲线如图 8.35 所示。

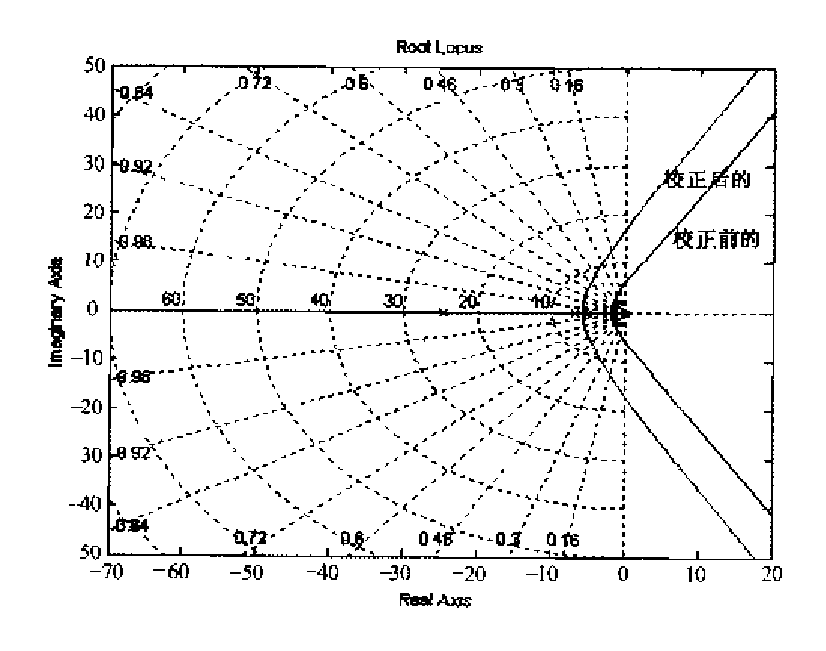


图 8.35　例 8-8 系统校正前后根轨迹图

由运行结果可知，串联超前校正环节的传递函数为 $G_c(s)=\dfrac{1.016s+1}{0.0404s+1}$。在阶跃响应图上单击鼠标右键，选择弹出菜单“Charateristics”，分别选择“Peak Response”、“Rise Time”、“Setting Time”，便可得到系统的超调、上升时间和调节时间。由运行图可知，校正前的系统超调为 $\sigma=17.3\%$，上升时间 $t_r=0.58\text{s}$，调节时间 $t_s=2.92\text{s}$；校正后的系统

超调为 $\sigma = 31.7\%$，上升时间 $t_r = 0.123\text{ s}$，调节时间 $t_s = 2.3\text{ s}$，可知校正后系统的性能提高了。

从根轨迹图可以看出，校正后系统根轨迹左移，从而提高系统的相对稳定性，缩短系统调节时间。

8.4.2　基于根轨迹法的滞后校正

用根轨迹法设计串联滞后校正的设计步骤为：

（1）绘制出未校正系统的根轨迹。

（2）根据要求的瞬态响应指标，确定希望的闭环主导极点，根据根轨迹的幅值条件，计算与主导极点对应的开环增益。

（4）按给定的性能指标中关于稳态误差的要求，计算应增大的误差系数值。

（5）由应增大的误差系数值确定校正装置 β 值，通常取 β 不超过 10。

（6）确定滞后校正装置的零点、极点。原则是使零点、极点靠近坐标原点，且两者相距 β 倍。

（7）绘出校正后系统的根轨迹，并求出希望的主导极点。

（8）由希望的闭环极点，根据幅值条件，适当调整放大器的增益。

（9）校验校正后系统各项性能指标，如不满足要求，则适当调整校正装置零点、极点。

【例 8-9】　已知系统开环传递函数如下：

$$G_o(s) = \frac{4}{s(s+2.5)}$$

试设计滞后校正环节，使其校正后系统的静态速度误差系数 $K_v \leqslant 6$，闭环主导极点满足阻尼比 $\zeta = 0.407$，并绘制校正前后系统的单位阶跃响应曲线、单位脉冲响应曲线和根轨迹。

解：计算串联滞后校正环节参数的子函数 MATLAB 程序代码如下。

```
function [Gc, kc]=zhjz_root(G, zeta, wc, Tz)
G=tf(G)
[r, k]=rlocus(G)
za=zeta/sqrt(1-zeta^2)
ri=r(1, find(imag(r(1, :))>0))
ra=imag(ri)./real(ri)
kc=spline(ra, k(find(imag(r(1, :))>0)), 1/za)
syms x
syms ng
syms dg
ng=poly2sym(G.num{1})
dg=poly2sym(G.den{1})
ess=limit(ng*kc/dg*x)
beta=round(100/sym2poly(ess)/wc)
```

```
Tp=Tz/beta
Gc=tf([1, Tz], [1, Tp])
```

主函数的代码如下：

```
num=4
den=conv([1, 0], [1, 2.5] )
G=tf(num, den)
zeta=0.407
wc=6
Tz=0.1
[Gc, Kc]=zhjz_root(G, zeta, wc, Tz)
GGc=G*Gc*Kc
Gy_close=feedhack(G, 1)
Gx_close=feedback(GGc, 1)
figure(1)
step(Gx_close, 'b')
hold on
step(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
figure(2)
impulse(Gx_close, 'b')
hold on
impulse(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
figure(3)
rlocus(G, GGc)
grid
gtext('校正前的')
gtext('校正后的')
```

运行结果如下：

```
%滞后校正环节传递函数
s + 0.1
---------
s + 0.025
%系统增益
```

```
Kc =
     2.3581
%原系统闭环传递函数
     4
---------------
s^2 + 2.5 s + 4
%校正后系统闭环传递函数
          9.433 s + 0.9433
---------------------------------
s^3 + 2.525 s^2 + 9.495 s + 0.9433
```

系统校正前后的单位阶跃响应曲线如图 8.36 所示。

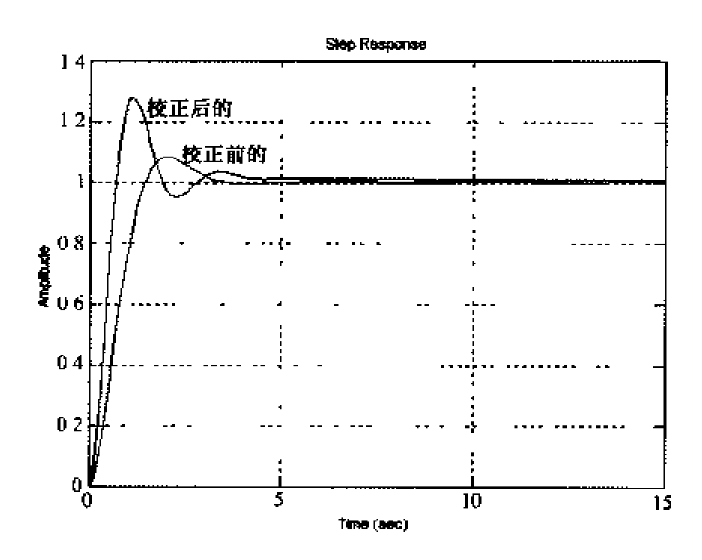


图 8.36　例 8-9 系统校正前后单位阶跃响应曲线

系统校正前后的单位脉冲响应曲线如图 8.37 所示。

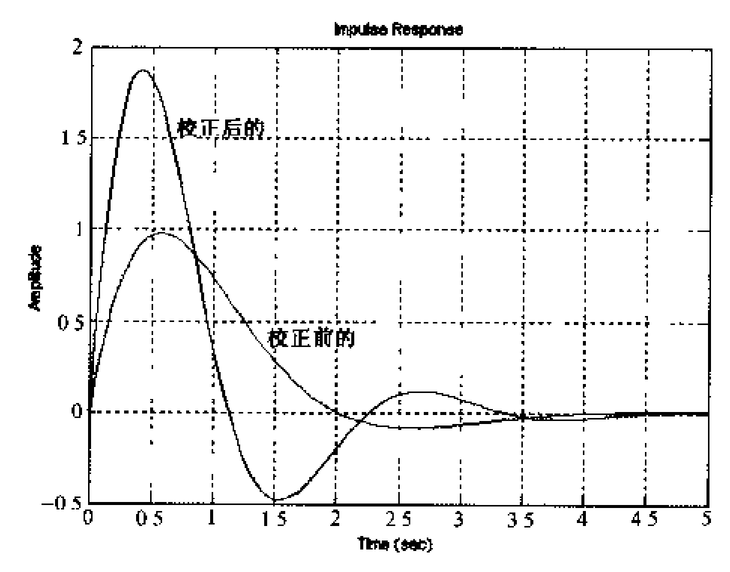


图 8.37　例 8-9 系统校正前后单位脉冲响应曲线

系统校正前后的根轨迹曲线如图 8.38 所示。

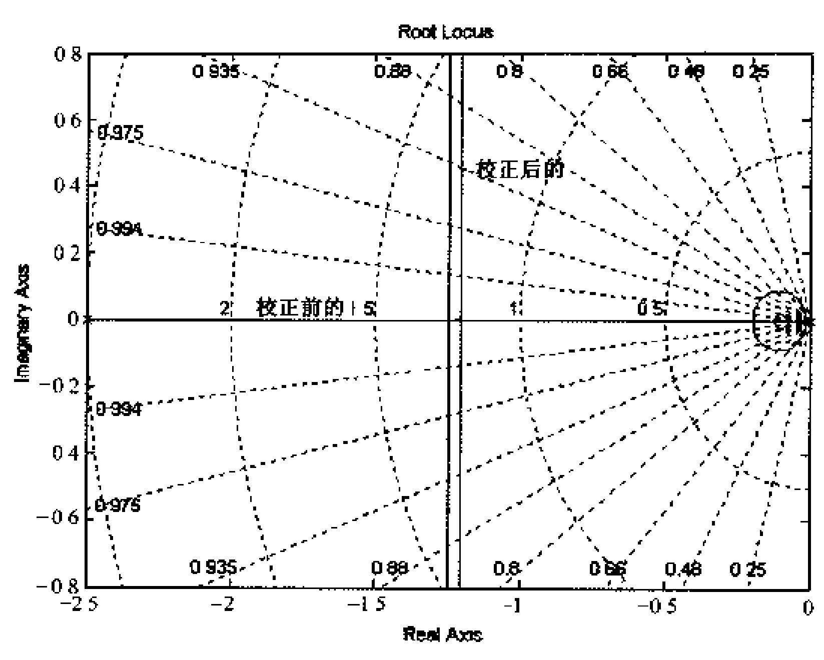


图 8.38　例 8-9 系统校正前后根轨迹

由运行结果可知，串联超前校正环节的传递函数为 $G_c(s)=\dfrac{s+0.1}{s+0.025}$。在阶跃响应图上单击鼠标右键，选择弹出菜单"Charateristics"，分别选择"Peak Response"、"Rise Time"、"Setting Time"，便可得到系统的超调、上升时间和调节时间。由运行图可知，校正前的系统超调为 $\sigma=8.08\%$，上升时间 $t_r=0.961\,\mathrm{s}$，调节时间 $t_s=2.99\,\mathrm{s}$；校正后的系统超调为 $\sigma=27.4\%$，上升时间 $t_r=0.478\,\mathrm{s}$，调节时间 $t_s=2.3\,\mathrm{s}$，可知校正后系统的性能提高了。

8.4.3　基于根轨迹法的超前滞后校正

用根轨迹法设计串联超前滞后校正的设计步骤为：

（1）根据要求的性能指标，确定希望的主导极点 s_d 的位置。

（2）为使闭环极点位于希望的位置，计算超前滞后校正中超前部分应产生的超前相角。

$$\varphi_c=\pm(2k+1)\pi-\angle G_o(s_d)$$

（3）超前滞后校正装置的传递函数为：

$$\varphi_c=\pm(2k+1)\pi-\angle G_o(s_d)$$

$$G_c(s)=K_c\left(\frac{s+\dfrac{1}{T_1}}{s+\dfrac{\beta}{T_1}}\right)\left(\frac{s+\dfrac{1}{T_2}}{s+\dfrac{1}{T_2\beta}}\right)$$

（4）对超前滞后校正中滞后部分的 T_2 选择要足够大，即

$$\left|\frac{s_d+\frac{1}{T_2}}{s_d+\frac{1}{T_2\beta}}\right|=1,\quad \left|\frac{s_d+\frac{1}{T_1}}{s_d+\frac{\beta}{T_1}}\right||K_1G_o(s_d)|=1,\quad \angle\left(s_d+\frac{1}{T_1}\right)-\angle\left(s_d+\frac{\beta}{T_1}\right)=\varphi$$

（5）利用求得的β值，选择T_2，使

$$\left|\frac{s_d+\frac{1}{T_2}}{s_d+\frac{1}{\beta T_2}}\right|\approx 1,\quad 0°<\angle\left(\frac{s_d+\frac{1}{T_2}}{s_d+\frac{1}{\beta T_2}}\right)<3°$$

（6）检验性能指标。

【例 8-10】　已知系统开环传递函数如下。

$$G_o(s)=\frac{8}{s(s+0.4)}$$

试设计超前滞后校正环节，使其校正后系统的静态速度误差系数$K_v\leqslant 5$，闭环主导极点满足阻尼比$\zeta=0.2$和自然振荡角频率$\omega_n=5\text{rad/s}$，相角裕度为 50°，并绘制校正前后系统的单位阶跃响应曲线、单位脉冲响应曲线和根轨迹。

解：函数 MATLAB 程序代码如下。

```
z=[]
p=[0, -0.4]
k=8
Gz=zpk(z, p, k)
G=tf(Gz)
zeta=0.2
wn=5
kc=1
Tz=0.1
dPm=50+5
ng=G.num{1}
dg=G.den{1}
[num, den]=ord2(wn, zeta)
s=roots(den)
s1=s(1)
Gc1=cqjz_root(G, s1, kc)
G1=G*Gc1*kc
[Gc2, Kc2]=zhjz_root(G, zeta, wn, Tz)
GGc=G1*Gc2*Kc2
Gy_close=feedback(G, 1)
Gx_close=feedback(GGc, 1)
```

```
figure(1)
step(Gx_close, 'b')
hold on
step(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
figure(2)
impulse(Gx_close, 'b')
hold on
impulse(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
figure(3)
rlocus(G, GGc)
grid
gtext('校正前的')
gtext('校正后的')
```

运行结果如下：

```
%超前校正环节传递函数
1.358 s + 1
-----------
0.425 s + 1
%滞后校正环节传递函数
s + 0.1
----------
s + 0.0125
%原系统闭环传递函数
          8
---------------
s^2 + 0.4 s + 8
%校正后系统闭环传递函数
                    1.358 s^2 + 1.136 s + 0.1
-------------------------------------------------
0.425 s^4 + 1.175 s^3 + 1.773 s^2 + 1.141 s + 0.1
```

系统校正前后的单位阶跃响应曲线如图 8.39 所示。

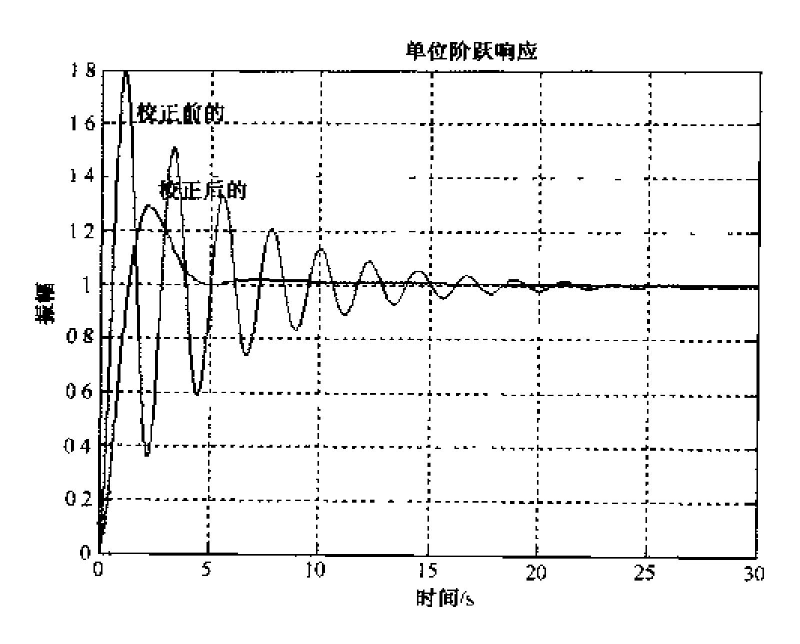


图 8.39　例 8-10 系统校正前后单位阶跃响应曲线

系统校正前后的单位脉冲响应曲线如图 8.40 所示。

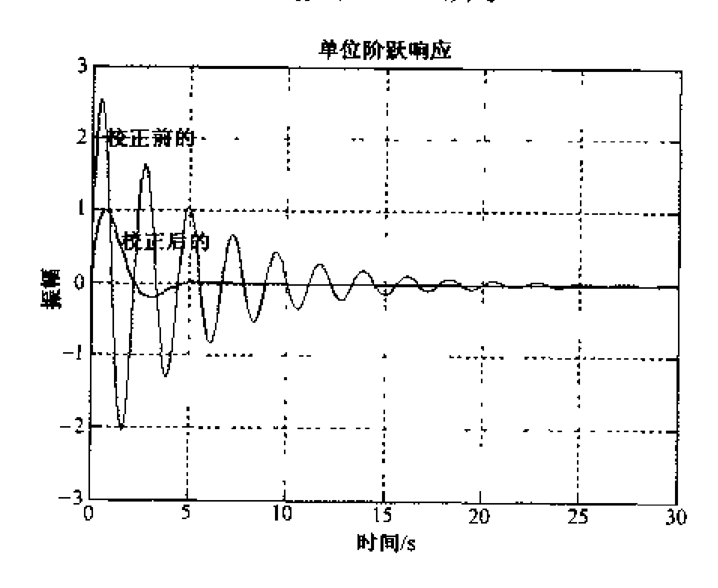


图 8.40　例 8-10 系统校正前后单位脉冲响应曲线

系统校正前后的根轨迹如图 8.41 所示。

由运行结果可知，串联超前校正环节的传递函数为 $G_c(s)=0.125\times\dfrac{s+0.1}{s+0.0125}\times\dfrac{1.358s+1}{0.425s+1}$。在阶跃响应图上单击鼠标右键，选择弹出菜单“Charateristics”，分别选择“Peak Response”、“Rise Time”和“Setting Time”，便可得到系统的超调、上升时间和调节时间。由运行图可知，校正前的系统超调为 $\sigma=79.9\%$，上升时间 $t_r=0.384\text{s}$，调节时间

$t_s=19.1\,\text{s}$；校正后的系统超调为 $\sigma=28.9\%$，上升时间 $t_r=0.907\,\text{s}$，调节时间 $t_s=4.19\,\text{s}$，可知校正后，系统的性能显著提高了。

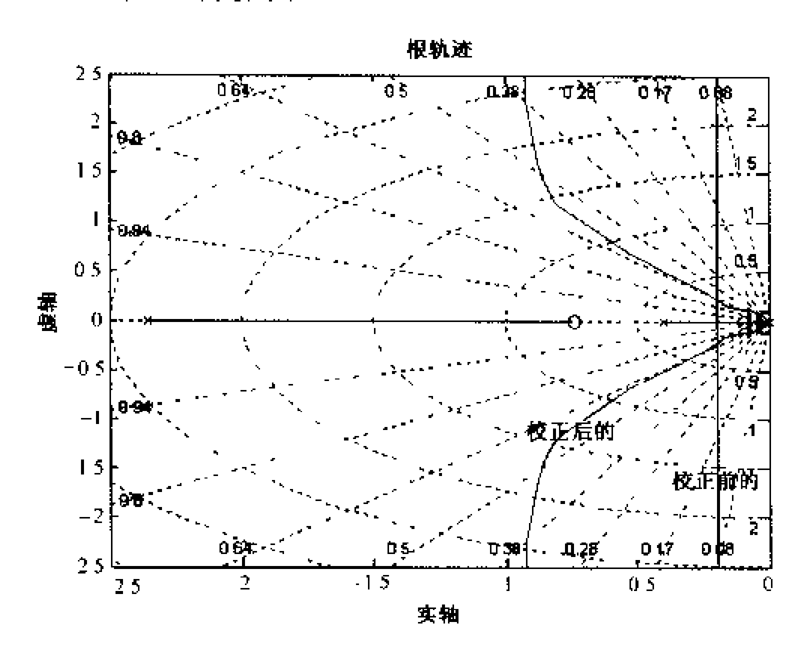


图 8.41　例 8-10 系统校正前后根轨迹

8.5　控制系统校正的频率响应法

前文提到对数频率特性的低频段影响系统的稳态误差，当要求系统的输出量以某一精度跟随输入时，需要系统在低频段具有相当的增益；在中频段，为了保证系统有足够的相位裕量，其特性斜率应为–20dB/dec，一般最大不超过–30dB/dec，而且在穿越频率附近要有一定的延伸段；为了减小高频干扰的影响，通常需要有尽快衰减的特性。

8.5.1　基于频率法的超前校正

频率法中的串联超前校正是利用校正装置的超前相位在穿越频率处对系统进行相位补偿，以提高系统的相位稳定裕量，同时也提高了穿越频率值，从而改善系统的稳定性和快速性。串联超前校正主要适用于稳定精度不需要改变，暂态性能不佳，而穿越频率附近相位变化平稳的系统。

应用频率法进行串联超前校正的步骤如下：

（1）根据所要求的稳态性能指标，确定系统的开环增益 K。

（2）绘制满足由(1)确定的值下的系统 Bode 图，并求出系统的相角裕量 γ_0。

（3）确定为使相角裕量达到要求值所需增加的超前相角 φ_c，即。

$$\varphi_c=\gamma-\gamma_0+\varepsilon$$

式中，γ 为要求的相角裕量，是考虑到校正装置影响剪切频率的位置而附加的相角裕量，当未校正系统中频段的斜率为–40dB/dec 时，取 $\varepsilon=5^\circ\sim15^\circ$，当未校正系统中频段斜率为–60dB/dec 时，取 $\varepsilon=5^\circ\sim20^\circ$。

（4）令超前校正网络的最大超前相角 $\varphi_m = \varphi_c$，则由下式求出校正装置的参数 α。

$$\alpha = \frac{1-\sin\varphi_m}{1+\sin\varphi_m}$$

（5）在 Bode 图上确定未校正系统幅值为 $20\lg\sqrt{\alpha}$ 时的频率 ω_m，该频率作为校正后系统的开环剪切频率 ω_c，即 $\omega_c = \omega_m$。

（6）由 ω_m 确定校正装置的转折频率 ω_1 和 ω_2。

$$\omega_1 = \frac{1}{\tau} = \omega_m\sqrt{\alpha}$$

$$\omega_2 = \frac{1}{\alpha\tau} = \frac{\omega_m}{\sqrt{\alpha}}$$

超前校正装置的传递函数为

$$G_c(s) = \alpha\frac{\tau s+1}{\alpha\tau s+1}$$

（7）将系统放大倍数增大 1/a 倍，以补偿超前校正装置引起的幅值衰减，即 $K_c = 1/a$。

（8）画出校正后系统的 Bode 图，校正后系统的开环传递函数如下。

$$G(s) = G_o(s)G_c(s)K_c$$

（9）校验系统的性能指标，若不满足要求，可增大 ε 值，从步骤（3）重新计算。

【例 8-11】　已知系统开环传递函数如下：

$$G_o(s) = \frac{2}{s(1+0.1s)(1+0.3s)}$$

试设计超前校正环节，使其校正后系统的静态速度误差系数 $K_v \leqslant 6$，相角裕度为 45°，并绘制校正前后系统的单位阶跃响应曲线，开环 Bode 图和闭环 Nyquist 图。

解：计算串联超前校正环节参数的子函数 MATLAB 程序代码如下。

```
function Gc=cqjz_frequency(G, kc, yPm)
G=tf(G)
[mag, pha, w]=bode(G*kc)
Mag=20*log10(mag)
[Gm, Pm,Wcg, Wcp]=margin(G*kc)
phi=(yPm-getfield(Pm, 'Wcg'))*pi/180
alpha=(1+sin(phi))/(1-sin(phi))
Mn=-10*log10(alpha)
Wcgn=spline(Mag, w, Mn)
T=1/Wcgn/sqrt(alpha)
Tz=alpha*T
Gc=tf([Tz, 1], [T, 1])
```

主函数 MATLAB 代码如下：

```
num=2
den=conv([1, 0], conv([0.3, 1], [0.1, 1]))
```

```
G=tf(num, den)
kc=3
yPm=45+12
Gc=cqjz_frequency(G, kc, yPm)
G=G*kc
GGc=G*Gc
Gy_close=feedback(G, 1)
Gx_close=feedback(GGc, 1)
figure(1)
step(Gx_close, 'b')
hold on
step(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
figure(2)
bode(G, 'b')
hold on
bode(GGc, 'r')
grid
gtext('校正前的')
gtext('校正后的')
gtext('校正前的')
gtext('校正后的')
figure(3)
nyquist(Gx_close, 'b')
hold on
nyquist(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
```

运行结果如下：

```
%超前校正环节传递函数
Transfer function:
0.3611 s + 1
-------------
0.09471 s + 1
%原系统闭环传递函数
```

```
         6
--------------------------
0.03 s^3 + 0.4 s^2 + s + 6
%校正后系统闭环传递函数
     2.167 s + 6
----------------------------------------------------
0.002841 s^4 + 0.06788 s^3 + 0.4947 s^2 + 3.167 s + 6
```

系统校正前后的单位阶跃响应曲线如图 8.42 所示。

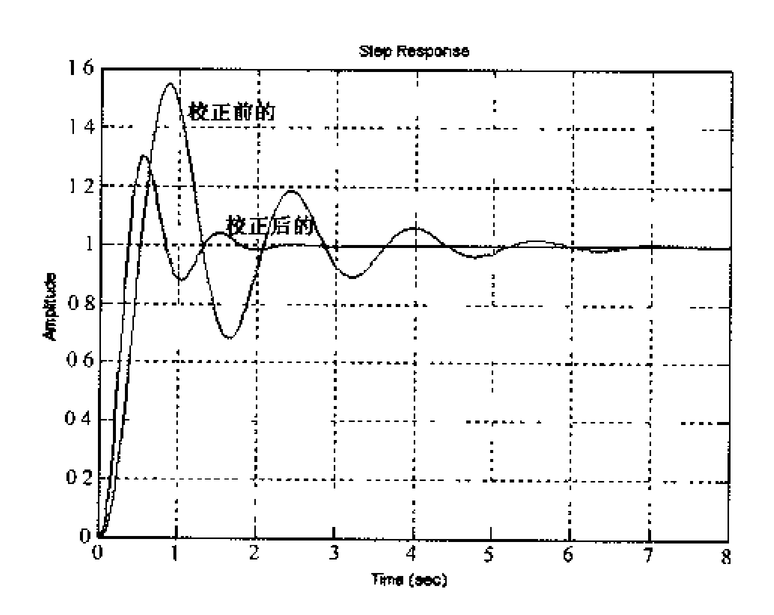


图 8.42　例 8-11 系统校止前后单位阶跃响应曲线

系统校正前后的开环 Bode 图如图 8.43 所示。

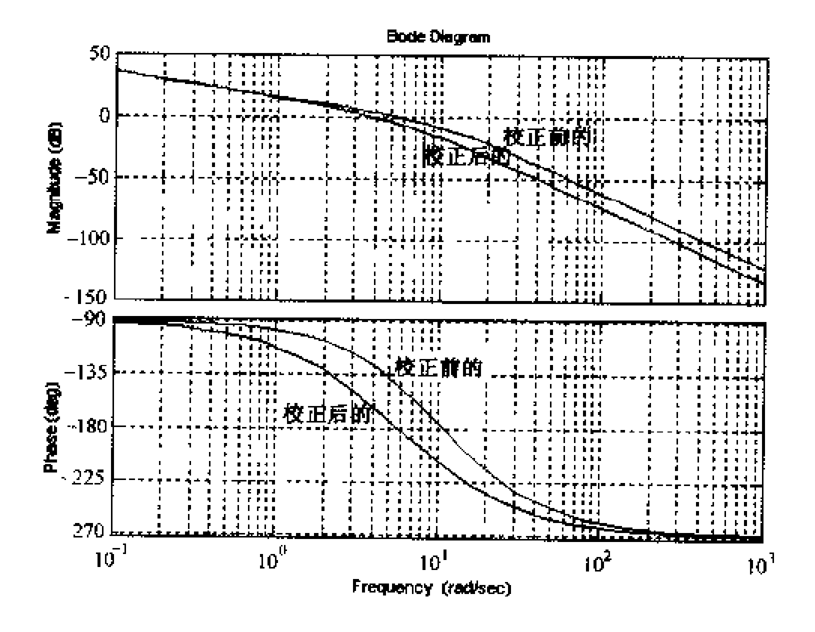


图 8.43　例 8-11 系统校正前后开环 Bode 图

系统校正前后的闭环 Nyquist 图如图 8.44 所示。

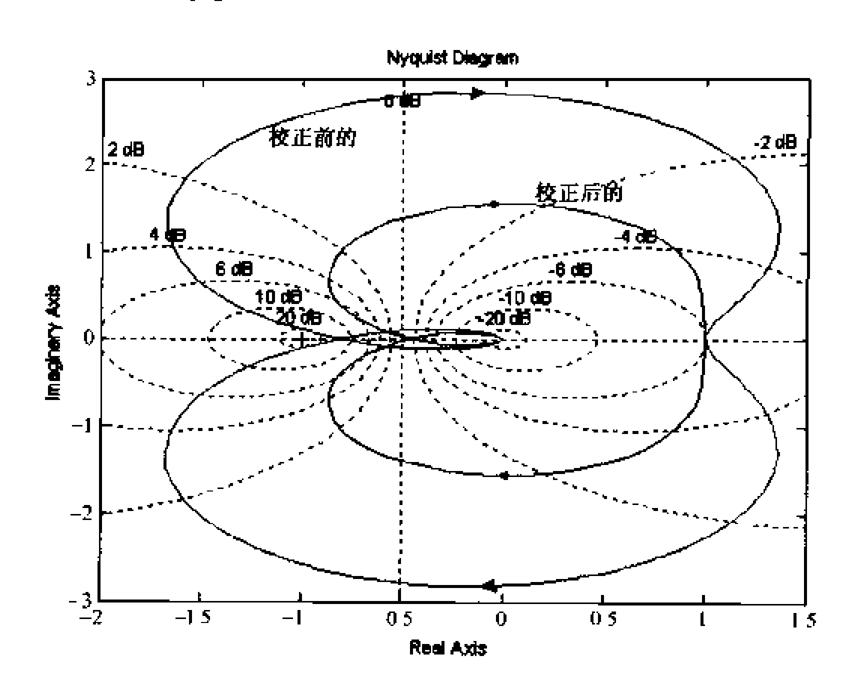


图 8.44　例 8-11 系统校正前后闭环 Nyquist 图

由运行结果可知，串联超前校正环节的传递函数为 $G_c(s)=\dfrac{0.3611s+1}{0.09471s+1}$。在 Bode 图上单击鼠标右键，选择弹出菜单“Charateristics”，选择“Stability”便可得到系统的幅值裕度、相角裕度和调节时间。由运行图可知，校正前的幅值裕度为 6.94 dB，相角裕度为 21.2；校正后的系统幅值裕度为 10.1 dB，相角裕度为 39，可知引入串联超前校正后，系统的带宽增加，闭环系统谐振峰值下降，静态误差系数增大。

8.5.2　基于频率法的滞后校正

频率法中的串联滞后校正在于提高系统的开环增益，改善控制系统的稳态性能，而尽量不影响原有系统的动态性能。串联滞后校正主要适用于未校正系统或经串联超前校正的系统的动态性能不能满足给定性能指标的需要，只需增大开环增益用以提高控制系统精度的一类系统中。

基于频率法的串联滞后校正步骤如下：

（1）根据稳态误差的要求确定系统开环放大系数，再用这一放大系数绘制原系统的 Bode 图，计算出本校正系统的相位裕量和增益裕量。

（2）根据给定相位裕量，增加 5°～15° 的补偿，估计需要附加的相角位移，找出符合这一要求的频率作为穿越频率 ω_c。

（3）确定出原系统在 $\omega=\omega_c$ 处幅值下降到零分贝时所必需的衰减量。使这一衰减量等于 $-20\lg\gamma_t$，从而确定 γ_t 的值。

（4）选择 $\omega_2=\dfrac{1}{T_d}$，计算 $\omega_1=\dfrac{\omega_2}{\gamma_t}$。

（5）计算校正后频率特性的相位裕量并判断是否满足给定要求，若不满足则重新计算。

（6）计算校正装置参数。

【例 8-12】　已知系统开环传递函数如下：

$$G_o(s)=\frac{2}{s(s+2.8)(s+0.8)}$$

试设计滞后校正环节，使其校正后系统的静态速度误差系数 $K_v \leqslant 6$，系统阻尼比 $\zeta=0.307$，并绘制校正前后系统的单位阶跃响应曲线，开环 Bode 图和闭环 Nyquist 图。

解：计算串联滞后校正环节参数的子函数 MATLAB 程序代码如下。

```
function Gc=cqjz_frequency(G, kc, dPm)
G=tf(G)
num=G.num{1}
den=G.den{1}
[mag, phase, w]=bode(G*kc)
wcg=spline(phase(1, :), w', dPm-180)
magdb=20*log10(mag)
Gr=-spline(w', magdb(1, :), wcg)
alpha=10^(Gr/20)
T=10/(alpha*wcg)
Gc=tf([alpha*T, 1], [T, 1])
```

主函数 MATLAB 代码如下：

```
num=2
den=conv([1, 0], conv([1, 2.8], [1, 0.8]))
G=tf(num, den)
zeta=0.307
Pm=2*sin(zeta)*180/pi
dPm=Pm+5
kc=2
Gc=zhjz_frequency(G, kc, dPm)
G=G*kc
GGc=G*Gc
Gy_close=feedback(G, 1)
Gx_close=feedback(GGc, 1)
figure(1)
step(Gx_close, 'b')
hold on
step(Gy_close, 'r')
grid
```

```
gtext('校正前的')
gtext('校正后的')
figure(2)
bode(G, 'b')
hold on
bode(GGc, 'r')
grid
gtext('校正前的')
gtext('校正后的')
gtext('校正前的')
gtext('校正后的')
figure(3)
nyquist(Gx_close, 'b')
hold on
nyquist(Gy_close, 'r')
grid
gtext('校正前的')
gtext('校正后的')
```

运行结果如下：

```
%滞后校止环节传递函数
16.08 s + 1
-----------
35.61 s + 1
%原系统闭环传递函数
              4
---------------------------
s^3 + 3.6 s^2 + 2.24 s + 4
%校正后系统闭环传递函数
           64.34 s + 4
-----------------------------------------------
35.61 s^4 + 129.2 s^3 + 83.37 s^2 + 66.58 s + 4
```

系统校正前后的单位阶跃响应曲线如图 8.45 所示。

系统校正前后的 Bode 图如图 8.46 所示。

系统校正前后的 Nyquist 图如图 8.47 所示。

由运行结果可知，串联滞后校正环节的传递函数为 $G_c(s)=\dfrac{16.08s+1}{35.61s+1}$。在 Bode 图上单击鼠标右键，选择弹出菜单“Charateristics”，再选择“Stability”便可得到系统的幅值裕度、相角裕度和调节时间。由运行图可知，校正前的幅值裕度为 6.09dB，相角裕度为 17.7；校正后的系统幅值裕度为 12.5dB，相角裕度为 36.4，系统的幅值裕度和相角裕度都得到了改善。

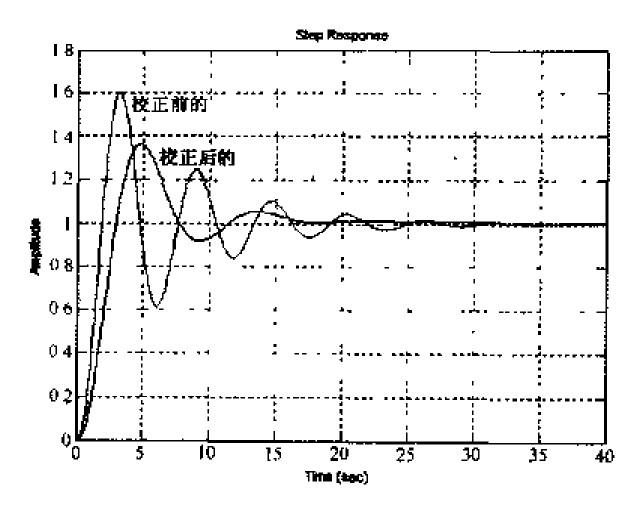


图 8.45　校正前后闭环系统的单位阶跃响应曲线

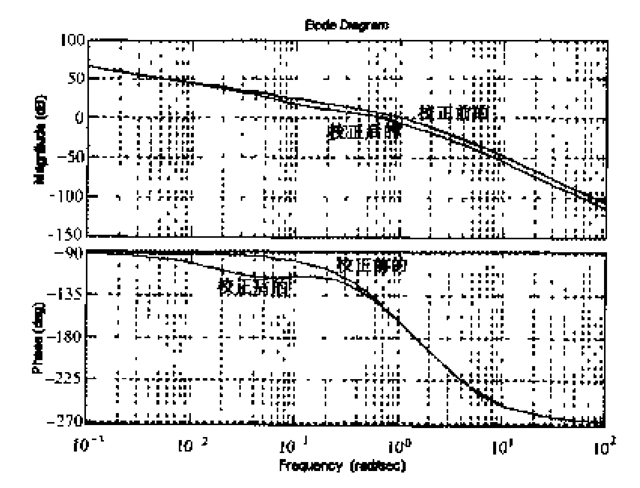


图 8.46　校正前后开环系统的 Bode 图

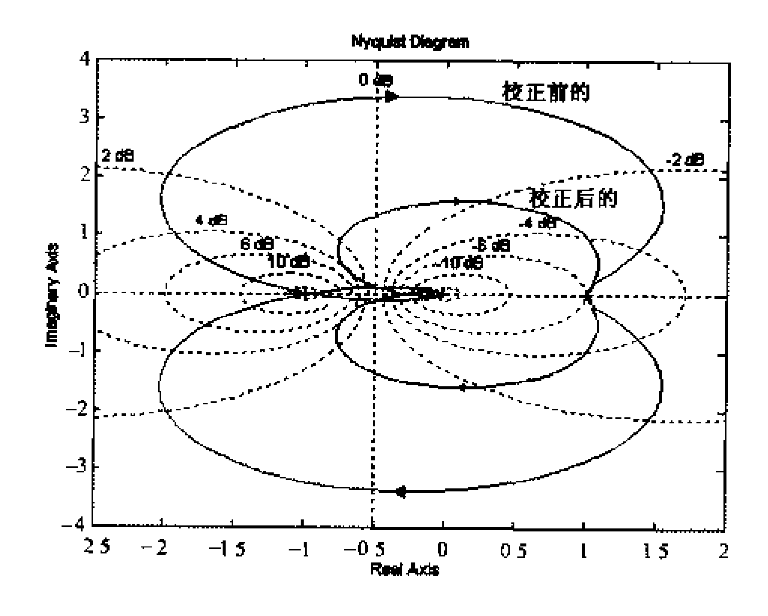


图 8.47　校正前后闭环系统的 Nyquist 图

频率法中的串联超前滞后校正，可兼有上述两种作用，主要用于要求较高但单纯的超前校正或滞后校正不能满足或无法应用的系统的校正，设计方法与前面类似，此处不再赘述。

第 9 章　线性系统状态空间分析

9.1　引　　言

在经典控制论中，常用高阶微分方程或传递函数来描述一个线性定常系统的运动规律，而微分方程或传递函数只能用于描述系统输入与输出之间的关系，不能描述系统内部的结构及其状态变量。

从经典控制论发展而来的现代控制论采用状态空间法来分析系统，用一组状态变量的一阶微分方程组作为系统的数学模型，它可反映出系统全部独立变量的变化情况，从而能同时确定系统的全部内部运动状态。

通过本章，读者对线性系统状态空间的基础知识和分析方法有一个全面的认识，并熟练使用 MATLAB 进行状态空间分析。

9.2　线性系统状态空间基础

9.2.1　状态空间基本概念

1. 状态

任何一个系统在特定时刻都有一个特定的状态，系统在 t_0 时刻的状态是 t_0 时刻的一种信息量，它与此后的输入一起惟一地确定系统在 $t \geqslant t_0$ 时的行为。

2. 状态变量

状态变量是一个完全表征系统时间域行为的最小内部变量组。

3. 状态向量

设系统有 n 个状态变量，用 $x_1(t), x_2(t), \cdots, x_n(t)$ 表示，而且把这些状态变量看做向量 $x(t)$ 的分量，则向量 $x(t)$ 称为状态向量，记为

$$x(t)=\left[x_1(t), x_2(t), \cdots, x_n(t)\right]' \tag{9-1}$$

4. 状态空间

以状态变量 $x_1(t), x_2(t), \cdots, x_n(t)$ 为轴的 n 维实向量空间称为状态空间。

5. 状态方程

描述系统状态变量与输入变量之间关系的一阶微分方程组（连续时间系统）或一阶差分方程组（离散时间系统）称为系统的状态方程，它表征了输入对内部状态的变换过程，其一般形式为：

$$\dot{x}(t) = f\left[x(t), u(t), t\right] \tag{9-2}$$

其中，t 是时间变量，$u(t)$ 是输入变量。

6. 输出方程

描述系统输出量与系统状态变量和输入变量之间函数关系的代数方程称为输出方程，它表征了系统内部状态变化和输入所引起的系统输出变换，是一个变化过程。输出方程的一般形式为：

$$y(t) = g\left[x(t), u(t), t\right] \tag{9-3}$$

7. 状态空间表达式

状态方程与输出方程的组合称为状态空间表达式，也称动态方程，它表征一个系统完整的动态过程，其一般形式为：

$$\begin{cases} \dot{x}(t) = f\left[x(t), u(t), t\right] \\ y(t) = g\left[x(t), u(t), t\right] \end{cases} \tag{9-4}$$

通常，对于线性定常系统，状态方程习惯写成如下形式：

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nr} \end{bmatrix} \cdot \begin{bmatrix} u_1 & u_2 & \cdots & u_r \end{bmatrix} \tag{9-5}$$

输出方程习惯写成如下形式：

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1r} \\ d_{21} & d_{22} & \cdots & d_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mr} \end{bmatrix} \cdot \begin{bmatrix} u_1 & u_2 & \cdots & u_r \end{bmatrix} \tag{9-6}$$

将其写成向量矩阵形式为：

$$\begin{cases} \dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u} \end{cases} \tag{9-7}$$

式中，

$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ 表示 n 维状态向量；

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$ 表示系统内部状态的系数矩阵，称为系统矩阵 $A_{n\times n}$；

$$B=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nr} \end{bmatrix}$$ 表示输入对状态作用的矩阵，称为输入（或控制）矩阵 $B_{n\times r}$；

$$C=\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}$$ 表示输出与状态关系的矩阵，称为输出矩阵 $C_{m\times n}$；

$$D=\begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1r} \\ d_{21} & d_{22} & \cdots & d_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mr} \end{bmatrix}$$ 表示输入直接对输出作用的矩阵，称为直接转移矩阵 $D_{m\times r}$，也称前馈系数矩阵。

A 由系统内部结构及其参数决定，体现了系统内部的特性，而 B 则主要体现了系统输入的施加情况，通常情况下 $D=0$。

式（9-7）表示的系统动态方程可用如图 9.1 所示的方框图表示。系统由两个前向通道和一个状态反馈回路组成，其中 D 通道表示控制输入 U 到系统输出 Y 的直接转移。

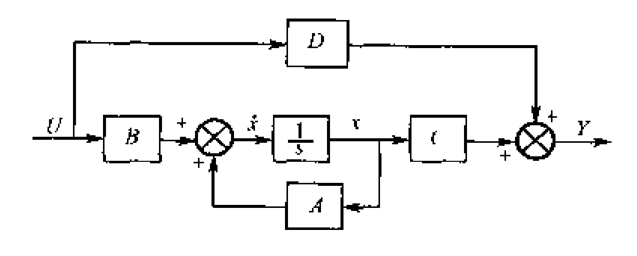


图 9.1　线性系统方框图

状态空间描述具有以下特点：

（1）状态空间描述考虑到了“输入—状态—输出”这一过程，考虑到了被经典控制理论的“输入—输出”描述所忽略的状态，因此它揭示了问题的本质，即输入引起状态的变化，而状态决定了输出。

（2）输入引起的状态变化是一个运动过程，数学上表现为向量微分方程，即状态方程。状态决定输出是一个变换过程，数学上表现为变换方程，即代数方程。

（3）系统的状态变量个数等于系统的阶数，一个 n 阶系统的状态变量个数为 n。

（4）对于给定的系统，状态变量的选择不惟一，状态变量的线性变换结果也可以作为状态变量。

（5）一般说来，状态变量不一定是物理上可测量或可观察的量，但从便于构造控制

系统来说，把状态变量选为可测量或可观察的量更合适。

9.2.2　状态空间实现

控制系统一般可分为电气、机械、机电、液压、热力等系统。要研究它们，一般先要建立其运动的数学模型（微分方程组、传递函数、动态方程等）。根据具体系统结构及其研究目的，选择一定的物理量作为系统的状态变量和输出变量，并利用各种物理定律，如牛顿定律、基尔霍夫电压电流定律、能量守恒定律等，建立系统的动态方程模型。

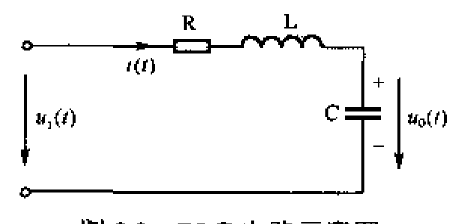


图 9.2　RLC 电路示意图

【例 9-1】　如图 9.2 所示的 RLC 电路中，系统的控制输入为电压 $u_{\mathrm{i}}(t)$，系统输出为电压 $u_{\mathrm{o}}(t)$，试建立系统的状态空间表达式。

解：建立系统状态方程的步骤如下。

（1）选择状态变量。

该 RLC 电路有两个独立的储能元件 L 和 C，可以取电容 C 两端电压 $u_{\mathrm{o}}(t)$ 和流过电感 L 的电流 $i(t)$ 作为系统的两个状态变量，分别记作 x_1 和 x_2。

（2）列写微分方程。

根据基尔霍夫电压定律和 R、L、C 元件的电压电流关系，可得到下列方程。

$$\begin{cases} L\dfrac{\mathrm{d}x_2(t)}{\mathrm{d}t}+R\cdot x_2(t)+x_1(t)=u_{\mathrm{i}}(t) \\ x_2(t)=C\dfrac{\mathrm{d}x_1(t)}{\mathrm{d}t} \end{cases}$$

（3）转化为状态变量的一阶微分方程组。

微分方程可整理为：

$$\begin{cases} \dot{x}_1=\dfrac{1}{C}x_2 \\ \dot{x}_2=-\dfrac{1}{L}x_1-\dfrac{R}{L}x_2+\dfrac{1}{L}u_{\mathrm{i}}(t) \\ y=x_1 \end{cases}$$

（4）把一阶微分方程组写成向量矩阵形式，即状态空间表达式。

一阶微分方程组写成矢量形式为：

$$\begin{cases} \dot{X}=\begin{bmatrix} 0 & \dfrac{1}{C} \\ -\dfrac{1}{L} & -\dfrac{R}{L} \end{bmatrix}X+\begin{bmatrix} 0 \\ \dfrac{1}{L} \end{bmatrix}u_{\mathrm{i}}(t) \\ Y=\begin{bmatrix} 1 & 0 \end{bmatrix}X \end{cases}$$

以上就是建立如图 9.2 所示 RLC 网络状态空间表达式的过程。

从经典控制理论中知道，任何一个线性系统都可以用下列线性微分方程表示：

$$\begin{aligned}&y^{(n)}(t)+a_{n-1}y^{(n-1)}(t)+\cdots+a_1y^{(1)}(t)+a_0y(t)\\&=b_mu^{(m)}(t)+b_{m-1}u^{(m-1)}(t)+\cdots+b_1u^{(1)}(t)+b_0u(t),\quad n\geqslant m\end{aligned} \tag{9-8}$$

式中，u 为系统的输入量，y 为系统的输出量，在零初始条件下，输入量与输出量的拉普拉斯变换之比，就是这个系统的传递函数：

$$G(s)=\frac{Y(s)}{U(s)}=\frac{b_ms^m+b_{m-1}s^{m-1}+\cdots+b_1s\ +b_0}{s^n+a_{n-1}s^{n-1}+\cdots+a_1s\ +a_0} \tag{9-9}$$

利用传递函数的概念，可以用以 s 为变量的代数方程表示系统的动态特性。如果传递函数分母中 s 的最高次数为 n，则称该系统为 n 阶系统。

传递函数只是表达了系统输出与输入的关系，没有表明系统内部的结构，而状态空间表达式可以完整地表明系统的内部结构。有了系统的状态空间表达式，就可以实现该系统。系统的实现一般有直接法、串联法和并联法。由系统的传递函数求其状态方程的过程称为系统的实现问题。

9.2.2.1 状态空间直接实现法

不失一般性，假设 $m=n$，则式（9-9）可以写成：

$$G(s)=\frac{Y(s)}{U(s)}=b_n+\frac{b'_{n-1}s^{n-1}+b'_{n-2}s^{n-2}+\cdots+b'_1s\ +b'_0}{s^n+a_{n-1}s^{n-1}+\cdots+a_1s\ +a_0} \tag{9-10}$$

其中，$b'_i=b_i-b_na_i\ \ (i=0,1,\cdots,n-1)$

令

$$\frac{Z(s)}{U(s)}=\frac{b'_{n-1}s^{n-1}+b'_{n-2}s^{n-2}+\cdots+b'_1s\ +b'_0}{s^n+a_{n-1}s^{n-1}+\cdots+a_1s\ +a_0} \tag{9-11}$$

代入式（9-10）得：

$$Y(s)=Z(s)+b_nU(s) \tag{9-12}$$

引入新变量 $Y_1(s)$，并且令

$$\frac{Y_1(s)}{U(s)}=\frac{1}{s^n+a_{n-1}s^{n-1}+\cdots+a_1s\ +a_0} \tag{9-13}$$

则由式（9-11）可得：

$$\frac{Z(s)}{Y_1(s)}=b'_{n-1}s^{n-1}+b'_{n-2}s^{n-2}+\cdots+b'_1\ s\ \ +b'_0 \tag{9-14}$$

将式（9-13）、式（9-14）分别作拉氏反变换，可得：

$$y_1^{(n)}+a_{n-1}y_1^{(n-1)}+\cdots+a_1y_1^{(1)}+a_0y_1=u(t) \tag{9-15}$$

$$z(t)=b'_{n-1}y_1^{(n-1)}+b'_{n-2}y_1^{(n-2)}+\cdots+b'_1\ y_1^{(1)}+b'_0\ y_1 \tag{9-16}$$

选择状态变量如下：

$$\begin{cases}x_1=y_1\\x_2=y_1^{(1)}=\dot{x}_1\\x_3=y_1^{(2)}=\dot{x}_2\\\quad\vdots\\x_n=y_1^{(n-1)}=\dot{x}_{n-1}\end{cases} \tag{9-17}$$

即：

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = x_4 \\ \quad\vdots \\ \dot{x}_n = y_1^{(n)} \end{cases} \tag{9-18}$$

则 $\dot{x}_n$ 为：

$$\begin{aligned} \dot{x}_n = y_1^{(n)} &= -a_0 y_1 - a_1 y_1^{(1)} - \cdots - a_{n-1} y_1^{(n-1)} + u(t) \\ &= -a_0 x_1 - a_1 x_2 - \cdots - a_{n-1} x_n + u(t) \end{aligned} \tag{9-19}$$

得系统状态方程为：

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = x_4 \\ \quad\vdots \\ \dot{x}_{n-1} = x_n \\ \dot{x}_n = -a_0 x_1 - a_1 x_2 - \cdots - a_{n-1} x_n + u(t) \end{cases} \tag{9-20}$$

对式（9-12）作拉氏反变换，并将式（9-16）代入，可得系统的输出 y 为：

$$y = z + b_n u = b_0' x_1 + b_1' x_2 + \cdots + b_{n-1}' x_n + b_n u \tag{9-21}$$

将式（9-20）和式（9-21）写成矢量形式，得到系统的动态方程为：

$$\begin{cases} \dot{\boldsymbol{X}} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -\alpha_0 & -\alpha_1 & -\alpha_2 & \cdots & -\alpha_{n-1} \end{bmatrix} \boldsymbol{X} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \boldsymbol{U} \\ \boldsymbol{Y} = \begin{bmatrix} b_0' & b_1' & b_2' & \cdots & b_{n-1}' \end{bmatrix} \boldsymbol{X} + b_n \boldsymbol{U} \end{cases} \tag{9-22}$$

式（9-22）所代表的系统实现的结构图如图 9.3 所示。这种系统的实现称做可控型（I 型）实现，关于可控型将在后续章节介绍。

注意：当式（9-10）中 $m<n$ 时，$b_n = 0$，$b_i' = b_i (i = 0, 1, \cdots, m)$，这时式（9-22）可以直接从传递函数的分子、分母多项式系数中写出。当式（9-10）中 $m=0$，即系统没有零点时，上述实现方法中系统状态变量就是输出变量的各阶导数 $y(0)$、$y(1)$、$\cdots$、$y(n-1)$。

在通常的低阶物理系统中，上述各状态变量的物理意义非常明确，如位移、速度、加速度等。

9.2.2.2　状态空间串联实现法

式（9-9）所示传递函数为多项式相除形式，分子多项式（num）为：

$$\text{num} = b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0 \tag{9-23}$$

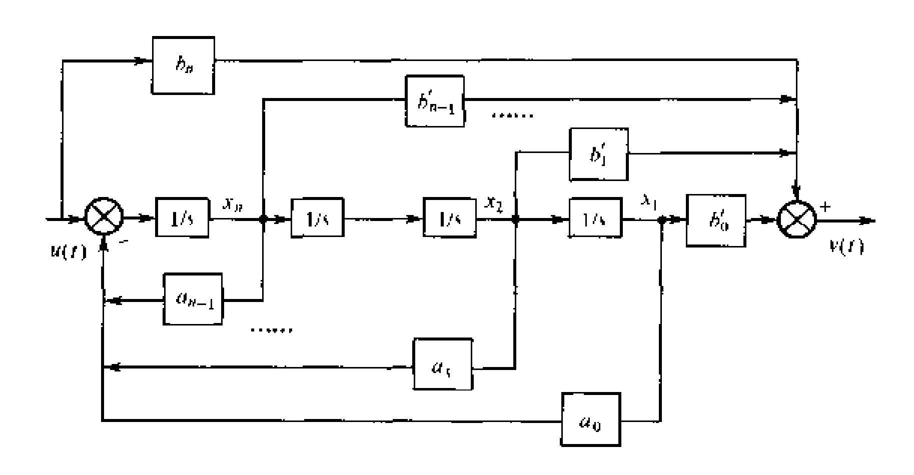


图 9.3　传递函数的直接法实现

分母多项式（den）为：

$$\text{den} = s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 \tag{9-24}$$

如果 $z_1, z_2, \cdots, z_m$ 为 $G(s)$ 的 m 个零点，$p_1, p_2, \cdots, p_n$ 为 $G(s)$ 的 n 个极点，那么 $G(s)$ 可以表示为：

$$\begin{aligned} G(s) &= \frac{b_m(s-z_1)(s-z_2)\ldots(s-z_m)}{(s-p_1)(s-p_2)\ldots(s-p_n)} \\ &= \frac{s-z_1}{s-p_1}\cdot\frac{s-z_2}{s-p_2}\cdot\cdots\cdot\frac{s-z_m}{s-p_m}\cdot\frac{b_m}{s-p_{m+1}}\cdot\cdots\cdot\frac{1}{s-p_n} \end{aligned} \tag{9-25}$$

所以系统的实现可以由 $\frac{s-z_1}{s-p_1}$、$\frac{s-z_2}{s-p_2}$，…，$\frac{1}{s-p_n}$ 共 n 个环节串联而成，如图 9.4 所示。

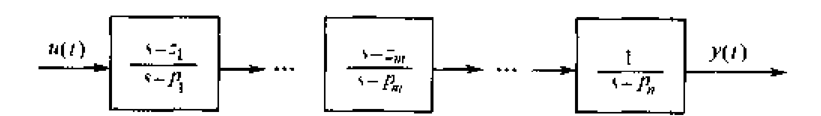


图 9.4　传递函数的串联实现结构图

图 9.4 对第一个环节，由于

$$\frac{s-z_1}{s-p_1} = 1 + \frac{p_1 - z_1}{s-p_1} = 1 + (p_1 - z_1)\frac{\frac{1}{s}}{1 - p_1\frac{1}{s}} \tag{9-26}$$

其结构图可用图 9.5 中虚框表示。其他环节可类似地等效变换，因此可以得到如图 9.5 所示的只有标准积分器、比例器、综合器组成的等效方框图。令各个积分器的输出为系统状态变量，则得系统状态方程为：

$$\begin{cases} \dot{x}_1 = p_1 x_1 + u \\ \dot{x}_2 = (p_1 - z_1)x_1 + u + p_2 x_2 = (p_1 - z_1)x_1 + p_2 x_2 + u \\ \vdots \\ \dot{x}_n = (p_1 - z_1)x_1 + (p_2 - z_2)x_2 + \cdots + (p_{n-1} - z_{n-1})x_{n-1} + p_n x_n + u \end{cases} \tag{9-27}$$

系统输出方程为：

$$y = b_m x_n = b_{n-1} x_n,\ (m = n-1) \tag{9-28}$$

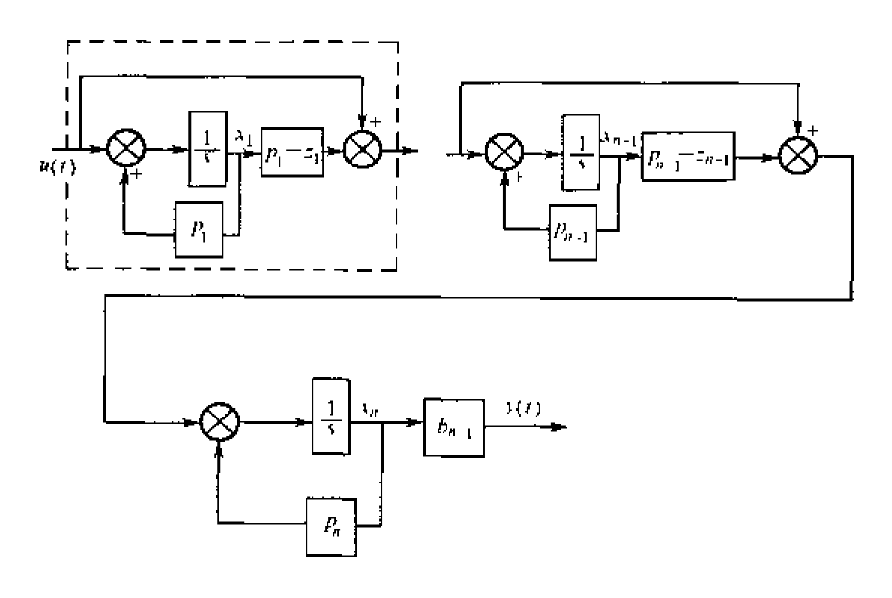


图 9.5　有重根的传递函数的串联实现结构图

写成矢量形式为：

$$\begin{cases} \dot{X} = \begin{bmatrix} p_1 & 0 & 0 & \cdots & 0 \\ p_1 - z_1 & p_1 & 1 & \cdots & 0 \\ p_1 - z_1 & p_2 - z_2 & p_2 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & 1 \\ p_1 - z_1 & p_2 - z_2 & p_3 - z_3 & \cdots & p_n \end{bmatrix} X + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} U \\ Y = \begin{bmatrix} 0 & 0 & 0 & \cdots & b_{n-1} \end{bmatrix} X \end{cases} \tag{9-29}$$

9.2.2.3　状态空间并联实现法

设系统传递函数为：

$$G(s) = \frac{\text{num}}{\text{den}} = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0} \tag{9-30}$$

式中，$m \leqslant n$。

$\text{den} = s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0 = 0$ 为系统的特征方程。当 $\text{den} = 0$ 有 n 个不等的特征根（$p_i, i = 1, 2, \cdots, n$）时，$G(s)$ 可以分解为 n 个分式之和，即：

$$G(s) = \frac{Y(s)}{U(s)} = \frac{c_1}{s - p_1} + \frac{c_2}{s - p_2} + \cdots + \frac{c_n}{s - p_n} \tag{9-31}$$

其中 $c_i = \lim\limits_{s \to p_i} (s - p_i) G(s)$，称做系统对应极点 p_i 的留数。

根据式（9-31），有：

$$Y(s)=\frac{c_1}{s-p_1}U(s)+\frac{c_2}{s-p_2}U(s)+\cdots+\frac{c_n}{s-p_n}U(s) \tag{9-32}$$

上式可以用如图 9.6 所示的并联方式实现，或用图 9.7（图 9.5 的等效形式）所示的方式实现。

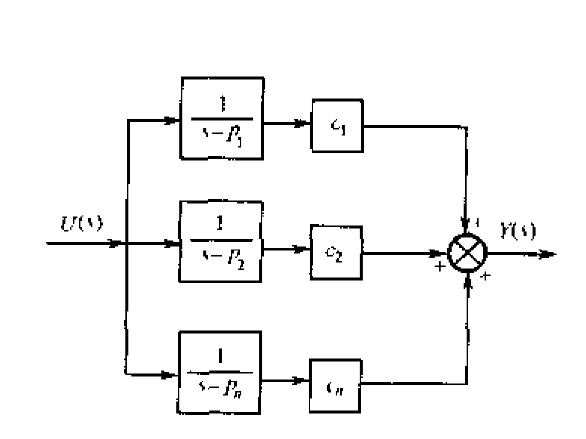



图 9.6　传递函数的并联实现结构图

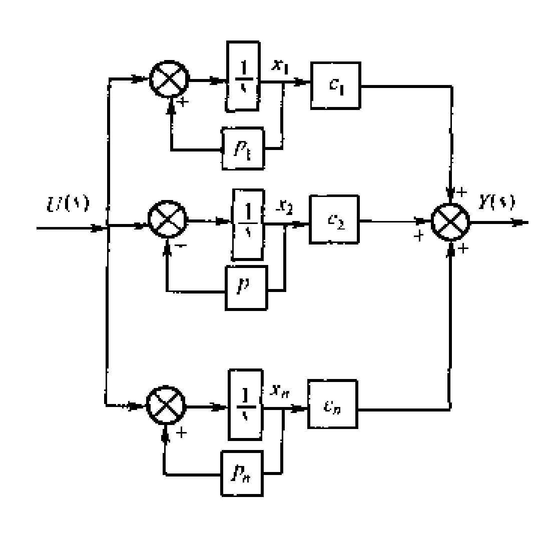


图 9.7　传递函数的并联实现结构等效图

从图 9.7 可得系统的状态方程为：

$$\begin{cases}\dot{x}_1=p_1x_1+u\\ \dot{x}_2=p_2x_2+u\\ \quad\vdots\\ \dot{x}_n=p_nx_n+u\end{cases} \tag{9-33}$$

输出方程为：

$$y=c_1x_1+c_2x_2+\cdots+c_nx_n \tag{9-34}$$

写成矢量形式为：

$$\begin{cases}\dot{X}=\begin{bmatrix}p_1 & 0 & \cdots & 0\\ 0 & p_2 & \cdots & 0\\ \vdots & \vdots & \cdots & \vdots\\ 0 & 0 & \cdots & p_n\end{bmatrix}X+\begin{bmatrix}1\\1\\ \vdots\\1\end{bmatrix}U\\ Y=\begin{bmatrix}c_1 & c_2 & \cdots & c_n\end{bmatrix}X\end{cases} \tag{9-35}$$

注意到这里的系统矩阵 A 为一个标准的对角型。

当上述 $G(s)$ 的分母 den = 0 有重根时，不失一般性地假设：

$$\mathrm{den}=(s-p_1)^q(s-p_{q+1})\cdots(s-p_n) \tag{9-36}$$

即 $s=p_1$ 为 q 重根，其他为单根。这时 $G(s)$ 可以分解为：

$$G(s)=\frac{\mathrm{num}}{\mathrm{den}}=\frac{c_{11}}{s-p_1}+\frac{c_{12}}{(s-p_1)^2}+\cdots+\frac{c_{1q}}{(s-p_1)^q}+\frac{c_{q+1}}{s-p_{q+1}}+\cdots+\frac{c_n}{s-p_n} \tag{9-37}$$

其中：

$$c_{1i}=\frac{1}{(q-i)!}\cdot\lim_{s\to p_1}\frac{\mathrm{d}^{q-i}}{\mathrm{d}s^{q-i}}\left[(s-p_1)^q G(s)\right],\ i=1,2,\cdots,q \tag{9-38}$$

$$c_j=\lim_{s\to p_j}\left[(s-p_j)G(s)\right],\quad j=q+1,q+2,\cdots,n \tag{9-39}$$

由式（9-37）可知：

$$G(s)=\frac{Y(s)}{U(s)}=\frac{c_{11}}{s-p_1}+\frac{c_{12}}{(s-p_1)^2}+\cdots+\frac{c_{1q}}{(s-p_1)^q}+\frac{c_{q+1}}{s-p_{q+1}}+\cdots+\frac{c_n}{s-p_n} \tag{9-40}$$

$$Y(s)=\frac{c_{11}}{s-p_1}U(s)+\frac{c_{12}}{(s-p_1)^2}U(s)+\cdots+\frac{c_{1q}}{(s-p_1)^q}U(s)+\frac{c_{q+1}}{s-p_{q+1}}U(s)+\cdots+\frac{c_n}{s-p_n}U(s) \tag{9-41}$$

式（9-41）可以用如图 9.7 所示的方框图表示。

取图 9.8 中每个积分器的输出为状态变量，则有：

$$\begin{cases}\dot{x}_1=p_1x_1+x_2\\ \dot{x}_2=p_2x_2+x_2\\ \quad\vdots\\ \dot{x}_{q-1}=p_1x_{q-1}+x_q\\ \dot{x}_q=p_1x_q+u\\ \quad\vdots\\ \dot{x}_n=p_nx_n+u\end{cases} \tag{9-42}$$

$$y=c_{1q}x_1+c_{1q-1}x_2+\cdots+c_{11}x_q+c_{q+1}x_{q+1}+\cdots+c_nx_n \tag{9-43}$$

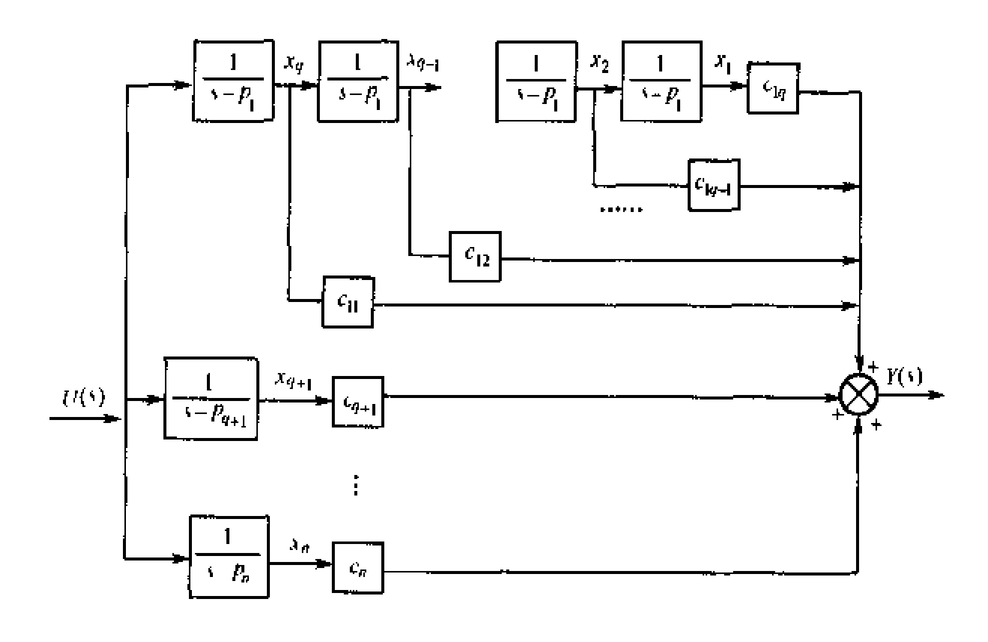


图 9.8　有重根传递函数的并联实现结构图

其矢量形式为：

$$\begin{cases} \dot{X} = \begin{bmatrix} p_1 & 1 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & p_1 & \cdots & \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & p_1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & & p_{q+1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & p_n \end{bmatrix} X + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} U \\ Y = \begin{bmatrix} c_{1q} & c_{1q-1} & \cdots & c_{11} & c_{q+1} & \cdots & c_n \end{bmatrix} X \end{cases} \tag{9-44}$$

注意：这里的 $\boldsymbol{A}$ 为约当标准型。关于约当标准型，请参见后续章节。

MATLAB 提供了建立状态空间模型的函数 tf2ss, zp2ss，它们的调用格式如下。

[A, B, C, D]=tf2ss(num, den)，其中，A, B, C, D 是状态空间模型的 4 个矩阵；num 是传递函数的分子多项式；den 是传递函数的分母多项式。

[A, B, C, D]=zp2ss(z, p, k)，其中，A, B, C, D 是状态空间模型的 4 个矩阵；z, p, k 分别是传递函数的零点、极点和增益。

sys=ss(sys1)，其中，sys1 是线性时不变系统 LTI 模型，sys 是状态空间模型。

【例 9-2】 已知系统的传递函数模型为 $G(s)=\dfrac{2s^2+8s+6}{s^3+8s^2+16s+6}$，求系统的状态空间模型。

解：MATLAB 程序代码如下。

```
%G(s)的分了多项式系数
num=[2, 8, 6]
%G(s)的分母多项式系数
den=[1, 8, 16, 6]
%由传递函数模型转换成状态空间模型
[A, B, C, D]=tf2ss(num, den)
```

运行结果如下：

```
A=
    -8   -16   -6
     1     0    0
     0     1    0
B=
     1
     0
     0
C=
     2     8    6
D=
     0
```

由运算结果可知，系统的状态空间表达式为：

$$\begin{cases}\dot{X}=\begin{bmatrix}-8 & -16 & -6\\ 1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}X+\begin{bmatrix}1\\0\\0\end{bmatrix}U\\ Y=\begin{bmatrix}2 & 8 & 6\end{bmatrix}X\end{cases}$$

【例 9-3】　已知系统的动力学微分方程为：

$$y^{(3)}(t)+3y^{(2)}(t)+3y^{(1)}(t)+y(t)=u^{(2)}(t)+2u^{(1)}(t)+u(t)$$

求系统的状态空间模型。

解：MATLAB 程序代码如下。

```
num=[1, 2, 1]
den=[1, 3, 3, 1]
%建立传递函数模型
sys1=tf(num, den)
%求状态空间表达式
sys=ss(sys1)
```

运行结果如下：

```
a=
            x1        x2       x3
    x1      -3     -0.75    -0.25
    x2       4         0        0
    x3       0         1        0
b=
         u1
    x1    1
    x2    0
    x3    0
c=
           x1     x2      x3
    y1      1    0.5    0.25
d=
         u1
    y1    0
Continuous-time model
```

由运算结果可知，系统的状态空间表达式为：

$$\begin{cases}\dot{X}=\begin{bmatrix}-3 & -0.75 & -0.25\\ 4 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}X+\begin{bmatrix}1\\0\\0\end{bmatrix}U\\ Y=\begin{bmatrix}1 & 0.5 & 0.25\end{bmatrix}X\end{cases}$$

【例 9-4】　已知系统的传递函数模型为 $G(s)=\dfrac{2(s+1)(s+3)}{s(s+2)(s+8)}$，求系统的状态空间模型。

解：MATLAB 程序代码如下。

```
%传递函数的零点
z=[-1, -3]
%传递函数的极点
p=[0, -2, -8]
%传递函数的增益
k=2
%由零极点增益模型转换成状态空间模型
[A, B, C, D]=zp2ss(z, p, k)
```

运行结果如下：

```
A=
      0      0      0
      1    -10     -4
      0      4      0
B=
      1
      0
      0
C=
    2.0000   -12.0000   -6.5000
D=
      0
```

由运算结果可知，系统的状态空间表达式为：

$$\begin{cases}\dot{X}=\begin{bmatrix}0 & 0 & 0\\1 & -10 & -4\\0 & 4 & 0\end{bmatrix}X+\begin{bmatrix}1\\0\\0\end{bmatrix}U\\ Y=\begin{bmatrix}2 & -12 & -6.5\end{bmatrix}X\end{cases}$$

9.2.3　状态空间模型描述

系统动态方程的建立，无论是从实际物理系统或系统方框图出发，还是从系统微分方程或传递函数出发，在状态变量的选取方面都有很大的人为的随意性，因而求得的系统状态方程也带有很大的人为因素和随意性，因此会得出不同的系统状态方程。实际物理系统虽然结构不可能变化，但状态变量取法不同就会产生不同的动态方程；系统方框图在取状态变量之前需要进行等效变换，而等效变换过程就有很大程度上的随意性，因此会产生一定程度上的结构差异，这也会导致动态方程差异的产生；从系统微分方程或

传递函数出发的系统实现问题，更是会导致迥然不同的系统内部结构，因而也肯定会产生不同的动态方程。所以说系统动态方程是非惟一的。

虽然同一实际物理系统、同一方框图、同一传递函数所产生的动态方程会各种各样，但其独立的状态变量的个数是相同的，而且各种不同动态方程间也有一定联系，这种联系就是变量间的线性变换关系。

如图 9.3 所示的传递函数的直接法实现，按照图上所示各状态变量的取法，有式（9-22）所示的动态方程。若将各变量的次序颠倒，即令

$$\begin{cases} \bar{x}_1 = x_n \\ \bar{x}_2 = x_{n-1} \\ \bar{x}_3 = x_{n-2} \\ \quad \vdots \\ \bar{x}_n = x_1 \end{cases} \tag{9-45}$$

即取

$$\bar{X} = \begin{bmatrix} 0 & 0 & \cdots & 1 \\ \vdots & & 1 & \vdots \\ \vdots & \iddots & \cdots & \vdots \\ 1 & \cdots & \cdots & 0 \end{bmatrix} X = TX \tag{9-46}$$

将 $X = T^{-1}X$ 代入式（9-22）所示的动态方程，可得：

$$\begin{cases} T^{-1}\bar{X} = AT^{-1}\bar{X} = +BU \\ Y = CT^{-1}\bar{X} \end{cases} \tag{9-47}$$

因此，系统的动态方程为：

$$\begin{cases} \dot{\bar{X}} = TAT^{-1}\bar{X} + TBU = \begin{bmatrix} -a_{n-1} & -a_{n-2} & \cdots & -a_0 \\ 1 & & & \vdots \\ & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} \bar{X} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} U \\ Y = CT^{-1}\bar{X} + b_n U = [b'_{n-1},\ b'_{n-2},\ \cdots,b'_1,\ b'_0] + b_n U \end{cases} \tag{9-48}$$

式（9-48）与式（9-22）是相同的。也就是说式（9-48）与式（9-22）代表的动态方程是一种线性变换的关系。

由于上述非奇异的变换矩阵 T 可以有无数种，所以系统的动态方程也有无数种。

虽然通过非奇异的线性变换可以求出无数种系统的动态方程，但是有几种标准型特别有用，如可控标准型、可观标准型、对角标准型和约当标准型。下面对最常见的对角标准型和约当标准型进行介绍。

9.2.3.1 对角标准型

设某系统的动态方程为：

$$\begin{cases} \dot{X} = AX + BU \\ Y = CX + DU \end{cases} \tag{9-49}$$

其中，系统矩阵 A 有 n 个不相等的特征根 $\lambda_i\ (i=1,2,3,\cdots,n)$，相应地有 n 个不相等的特征向量 $m_i\ (i=1,2,3,\cdots,n)$，因此矩阵 A 的特征矩阵（模态矩阵）为 $M=\begin{bmatrix} m_1 & m_2 & \cdots & m_n \end{bmatrix}$。

利用矩阵论知识可得：

$$A'=M^{-1}AM=\begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}=\mathrm{diag}[\lambda_1,\lambda_2,\ \cdots,\lambda_n] \tag{9-50}$$

对代表原系统的式（9-49）进行下列线性变换：

$$X=MZ \tag{9-51}$$

得：

$$\begin{cases} M\dot{Z}=AMZ+BU \\ Y=CMZ+DU \end{cases} \tag{9-52}$$

式（9-52）可写成：

$$\begin{cases} \dot{Z}=M^{-1}AMZ+M^{-1}BU=A'Z+B'U \\ Y=C'Z+D'U \end{cases} \tag{9-53}$$

其中，

$$\begin{aligned} A'&=M^{-1}AM=\mathrm{diag}[\lambda_1,\lambda_2,\ \cdots,\lambda_n] \\ B'&=M^{-1}B \\ C'&=CM \\ D'&=D \end{aligned} \tag{9-54}$$

这样就将式（9-49）转化成了式（9-53）所示的对角型。从上面各式可以看出，只要求出系统矩阵 A 的 n 个不同特征根 $\lambda_i\ (i=1,2,3,\cdots,n)$，就可以直接写出 A' 和 D'，但要求出 B' 和 C'，还需根据矩阵论知识求出矩阵 M 及其逆矩阵 M^{-1}，然后根据式（9-54）才能求得。

将系统矩阵 A 变换为标准对角型，其变换矩阵也是非惟一的，实际上有无数种。这无数种变换矩阵不会改变式（9-54）中 A' 的对角型形式，只会改变 B' 和 C' 的结果。

另外，还有一种不同形式的标准对角型状态空间表达式，它的系统矩阵 A' 与式（9-54）一样，并且此时 B' 也有标准的形式 $(1,1,\cdots,1)^{\mathrm{T}}$。

要得到上述标准型，只需进行如下线性变换：

$$X=MTZ \tag{9-55}$$

其中，M 为模态矩阵，T 为一个待定的对角矩阵，设 $T=\mathrm{diag}(t_1,t_2,\cdots,t_n)$。此时，式（9-53）变为：

$$\begin{cases} \dot{Z}=T^{-1}M^{-1}AMTZ+T^{-1}M^{-1}BU=A'Z+B'U \\ Y=CMTZ+DU \end{cases} \tag{9-56}$$

其中，

$$\begin{aligned} A'&=T^{-1}M^{-1}AMT\text{ [illegible] }[\lambda_1,\lambda_2,\ \cdots,\lambda_n] \\ B'&=T^{-1}M^{-1}B=\text{ [illegible] }\cdots\ 1)^{\mathrm{T}} \\ C'&=CMT\quad\text{[illegible]} \\ D'&=D\quad\text{[illegible]} \end{aligned} \tag{9-57}$$

T 矩阵可以通过下式求得：

$$M^{-1}B = T(1,1,\cdots,1)^{\mathrm{T}} = \mathrm{diag}(t_1,t_2,\cdots,t_n)(1,1,\cdots,1)^{\mathrm{T}} \tag{9-58}$$

MATLAB 提供了函数 canon(A, B, C, D, 'mod')可以将系统直接转化为对角型。运行结果返回 As, Bs, Cs, Ds 为对角型，返回的 Ts 表示所作的线性变换。

【例 9-5】 已知系统的状态空间模型为 $\begin{cases}\dot{X} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -6 & -11 & -6\end{bmatrix}X + \begin{bmatrix}1\\0\\0\end{bmatrix}U \\ Y = \begin{bmatrix}1 & 1 & 0\end{bmatrix}X\end{cases}$，求系统的对角标准型。

解：MATLAB 程序代码如下。

```
A=[0 1 0; 0 0 1; -6 -11 -6];
B=[1; 0; 0];
C=[1 1 0];
D=0;
%生成对角标准型
[As, Bs, Cs, Ds, Ts]=canon(A, B, C, D, 'mod')
```

运行结果如下：

```
As=
    -1.0000         0         0
          0   -2.0000         0
          0         0   -3.0000
Bs=
    -5.1962
   -13.7477
    -9.5394
Cs=
     0.0000   -0.2182    0.2097
Ds=
      0
Ts=
    -5.1962   -4.3301   -0.8660
   -13.7477  -18.3303   -4.5826
    -9.5394  -14.3091   -4.7697
```

由运算结果可知系统的对角标准型为：

$$\begin{cases}\dot{X} = \begin{bmatrix}-1 & 0 & 0\\ 0 & -2 & 0\\ 0 & 0 & -3\end{bmatrix}X + \begin{bmatrix}-5.1962\\ -13.7477\\ -9.5394\end{bmatrix}U \\ Y = \begin{bmatrix}0 & -0.2182 & 0.2097\end{bmatrix}X\end{cases}$$

9.2.3.2 约当标准型

设系统有 k 个 m_i 重特征值 $\lambda_i\ (i=1,2,3,\cdots,k)$，那么其约当标准型为：

$$\begin{cases}\dot{Z}=JZ+\tilde{B}U\\Y=\tilde{C}Z+\tilde{D}U\end{cases}\tag{9-59}$$

其中，J 为约当矩阵，即

$$J=\mathrm{diag}\begin{bmatrix}J_1 & J_2 & \cdots & J_k\end{bmatrix}\tag{9-60}$$

J_i 为 m_i 重特征根 λ_i 所对应的约当块，即

$$J_i=\begin{bmatrix}\lambda_i & 1 & & 0\\ & \lambda_i & \ddots & \\ & & & 1\\ 0 & & & \lambda_i\end{bmatrix}_{(m_i\times m_i)}\tag{9-61}$$

设现有系统的动态方程为：

$$\begin{cases}\dot{X}=AX+BU\\Y=CX+DU\end{cases}\tag{9-62}$$

求线性变换矩阵 T_J，使得式（9-62）经变换后得到式（9-59）所示的约当标准型。要得到上述标准型，只需作线性变换：

$$X=T_JZ\tag{9-63}$$

代入式（9-62）得：

$$\begin{cases}T_J\dot{Z}=AT_JZ+BU\\Y=CT_JZ+DU\end{cases}\tag{9-64}$$

即

$$\begin{cases}\dot{Z}=T_J^{-1}AT_JZ+T_J^{-1}BU\\Y=CT_JZ+DU\end{cases}\tag{9-65}$$

对照式（9-59）约当标准型，有：

$$T_J^{-1}AT_J=J\tag{9-66}$$

式（9-66）可写成：

$$AT_J=T_JJ\tag{9-67}$$

设

$$T_J=[t_1,t_2,\cdots,t_n]\tag{9-68}$$

代入式（9-67）得：

$$[t_1,t_2,\cdots,t_n]J=A[t_1,t_2,\cdots,t_n]\tag{9-69}$$

式（9-69）可写成：

$$[t_1,t_2,\cdots,t_n]\begin{bmatrix}J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_k\end{bmatrix}=A[t_1,t_2,\cdots,t_n]\tag{9-70}$$

对于 m_i 重的特征根 λ_i ，$\boldsymbol{T}_J$ 中有 m_i 个列向量 $\left[t_1, t_2, \cdots, t_{m_i}\right]$ 与 $\boldsymbol{J}_i$ 对应，即

$$\left[t_1, t_2, \cdots, t_{m_i}\right]\begin{bmatrix} \lambda_i & 1 & & 0 \\ & \lambda_i & \ddots & \\ & & & 1 \\ 0 & & & \lambda_i \end{bmatrix} = \boldsymbol{A}\left[t_1, t_2, \cdots, t_{m_i}\right] \tag{9-71}$$

式（9-71）展开即得：

$$\begin{cases} \left|\lambda_i \boldsymbol{I} - \boldsymbol{A}\right| t_1 = 0 \\ \left|\lambda_i \boldsymbol{I} - \boldsymbol{A}\right| t_1 = -t_1 \\ \quad\vdots \\ \left|\lambda_i \boldsymbol{I} - \boldsymbol{A}\right| t_{m_i} = -t_{m_i - 1} \end{cases} \tag{9-72}$$

由式（9-72）即可求得各特征根 λ_i 所对应的 m_i 个列向量 $(t_1, t_2, \cdots, t_{m_i})$，从而求得变换矩阵 $\boldsymbol{T}_J$，进一步根据式（9-65）即可求得系统的约当标准型。

【例 9-6】 已知系统的传递函数模型为 $G(s)=\dfrac{2s+1}{s^3+7s^2+14s+8}$，求系统的约当标准型。

解：MATLAB 程序代码如下。

```
num=[2, 1]
den=[1, 7, 14, 8]
%求系统的分式表达式
[r, p, k]=residue(num, den)
A=diag(p)
B=ones(length(r), 1)
C=rat(r)
D=0
```

运行结果如下：

```
A=
    -4.0000         0         0
          0   -2.0000         0
          0         0   -1.0000
B=
     1
     1
     1
C=
-1 + 1/(-6)
2 + 1/(-2)
-0 + 1/(-3)
D=
     0
```

由运算结果可知，系统的约当标准型为：

$$\begin{cases} \dot{X}=\begin{bmatrix} -4 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -1 \end{bmatrix} X+\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} U \\ Y=\begin{bmatrix} -\frac{7}{6} & \frac{3}{2} & -\frac{1}{3} \end{bmatrix} X \end{cases}$$

【例 9-7】 已知系统的传递函数模型为$G(s)=\dfrac{2s^2+5s+3}{(s-1)^3}$，求系统的约当标准型。

解：MATLAB 程序代码如下。

```
num=[2, 5, 3]
den=conv( [1, -1], conv( [1, -1], [1, -1]) )
%求系统的分式表达式
[r, p, k]=residue(num, den)
A=diag(p)
B=rot90(r)
C=ones(1, length(r))
D=0
```

运行结果如下：

```
A=
     1.0000         0         0
          0    1.0000         0
          0         0    1.0000
B=
     2.0000    9.0000   10.0000
C=
     1     1     1
D=
     0
```

由运算结果可知，系统的约当标准型为：

$$\begin{cases} \dot{X}=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} X+\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} U \\ Y=\begin{bmatrix} 2 & 9 & 10 \end{bmatrix} X \end{cases}$$

9.2.4 状态方程求解

9.2.4.1 线性连续定常齐次方程求解

所谓齐次方程解也就是系统的自由解，是系统在没有控制输入的情况下，由系统的初始状态引起的自由运动，其状态方程为：

$$\dot{X} = AX\text{，其中 } X\big|_{t=0} = X_0 \tag{9-73}$$

式中，X 是 $n\times1$ 维的状态向量，A 是 $n\times n$ 维的常数矩阵。

标量定常微分方程 $\dot{x} = ax$ 的解为：

$$x = e^{at}x(0) = \left[1 + at + \frac{1}{2!}a^2t^2 + \cdots + \frac{1}{k!}a^kt^k + \cdots\right]x(0) \tag{9-74}$$

与式（9-74）类似，假设式（9-73）的解 $X(t)$ 为时间 t 的幂级数形式，即

$$X(t) = b_0 + b_1t + \cdots + b_kt^k + \cdots \tag{9-75}$$

其中，$b_i(i = 0, 1, \cdots)$ 为与 $X(t)$ 同维的矢量。

将式（9-75）两边对 t 求导，并代入式（9-73），得：

$$\begin{aligned} b_1 + 2bt + \cdots + kb_kt^{k-1} + \cdots &= A(b_0 + b_1t + \cdots + b_kt^k + \cdots) \\ &= Ab_0 + Ab_1t + \cdots + Ab_kt^k + \cdots \end{aligned} \tag{9-76}$$

式（9-76）对任意时间 t 都应该成立，所以变量 t 的各阶幂的系数都应该相等，即

$$\begin{cases} b_1 = Ab_0 \\ 2b_2 = Ab_1 \\ \quad\vdots \\ kb_k = Ab_{k-1} \\ \quad\vdots \end{cases} \tag{9-77}$$

即：

$$\begin{cases} b_1 = Ab_0 \\ b_2 = \dfrac{1}{2}Ab_1 = \dfrac{1}{2}A^2b_0 \\ \quad\vdots \\ b_k = \dfrac{1}{k}Ab_{k-1} = \dfrac{1}{k!}A^kb_0 \\ \quad\vdots \end{cases} \tag{9-78}$$

将系统初始条件 $X(t)\big|_{t=0} = X_0$ 代入式（9-75）可得 $b_0 = X_0$，代入式（9-78）可得：

$$\begin{cases} b_1 = AX_0 \\ b_2 = \dfrac{1}{2}A^2X_0 \\ \quad\vdots \\ b_k = \dfrac{1}{k!}A^kX_0 \\ \quad\vdots \end{cases} \tag{9-79}$$

将式（9-79）代入式（9-75）可得式（9-73）的解：

$$X(t) = (I + At + \frac{1}{2}A^2t^2 + \cdots + \frac{1}{k!}A^kt^k + \cdots)X_0 \tag{9-80}$$

记为：

$$e^{At}=I+At+\frac{1}{2}A^2t^2+\cdots+\frac{1}{k!}A^kt^k+\cdots \tag{9-81}$$

其中，e^{At} 为一矩阵指数函数，它是一个 $n\times n$ 方阵。所以式（9-80）变为：

$$X(t)=e^{At}X_0 \tag{9-82}$$

当式（9-73）给定的是 t_0 时刻的状态值 $X(t_0)$ 时，不难证明：

$$X(t)=e^{A(t-t_0)}X_{t_0} \tag{9-83}$$

从式（9-83）可看出，$e^{A(t-t_0)}$ 形式上是一个矩阵指数函数，也是一个各元素随时间 t 变化的 $n\times n$ 矩阵。但本质上，它的作用是将 t_0 时刻的系统状态矢量 $X(t_0)$ 转移到 t 时刻的状态矢量 $X(t)$，也就是说它起到了系统状态转移的作用，因此称之为状态转移矩阵（The State Transition Matrix），记为：

$$\Phi(t-t_0)=e^{A(t-t_0)} \tag{9-84}$$

因此 $X(t)$ 的解为：

$$X(t)=\Phi(t-t_0)X(t_0) \tag{9-85}$$

9.2.4.2 矩阵指数的性质及求法

对线性定常系统来说，齐次方程的求解可归结为求矩阵指数 $e^{A(t-t_0)}$。$e^{A(t-t_0)}$ 具有以下基本性质：

（1）性质 1：组合性质，从 $-\tau$ 转移到 0，再从 0 转移到 t 的组合等于从 $-\tau$ 转移到 t，即 $e^{A\tau}\cdot e^{At}=e^{A(t+\tau)}$。

（2）性质 2：状态矢量从时刻 t 转移到时刻 t，状态矢量不变，$e^{A0}=I$，$e^{At}\cdot e^{-At}=I$。

（3）性质 3：状态转移矩阵的逆意味着时间的逆转，$\left[e^{At}\right]^{-1}=e^{-At}$。

（4）性质 4：若矩阵 A，B 可交换，即 $AB=BA$，那么 $e^{(A+B)t}=e^{At}\cdot e^{Bt}$，否则不成立。

（5）性质 5：$\frac{de^{At}}{dt}=Ae^{At}=e^{At}\cdot A$，该性质可用来从给定的 e^{At} 矩阵中求出系统矩阵 A，即：

$$A=\left[e^{At}\right]^{-1}\cdot\frac{de^{At}}{dt}=e^{-At}\cdot\frac{de^{At}}{dt} \tag{9-86}$$

（6）性质 6：若矩阵 A 为一对角阵，即 $A=\mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_n)$，那么 e^{At} 也是对角阵，且 $e^{At}=\mathrm{diag}(e^{\lambda_1 t},e^{\lambda_2 t},\cdots,e^{\lambda_n t})$。

（7）性质 7：若 $n\times n$ 方阵 A 有 n 个不相等的特征根 $\lambda_i(i=1,2,\cdots,n)$，$M$ 是 A 的模态矩阵，$\tilde{A}=\mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_n)$，则 $M^{-1}e^{At}M=e^{\tilde{A}t}$，即 e^{At} 为：

$$e^{At}=Me^{\tilde{A}t}M^{-1}\ e^{At} \tag{9-87}$$

式（9-87）常用来求 e^{At}。

（8）性质 8：若 J_i 为 $m_i\times m_i$ 阶约当块，即 $J_i=\begin{bmatrix}\lambda_i & 1 & & 0\\ & \lambda_i & \ddots & \\ & & & 1\\ 0 & & & \lambda_i\end{bmatrix}_{(m_i\times m_i)}$，则

$$
e^{J_i t} = e^{\lambda_i t}\begin{bmatrix} 1 & t & \dfrac{t^2}{2} & \cdots & \dfrac{t^{m_i-1}}{(m_i-1)!} \\ & 1 & t & \cdots & \vdots \\ & & 1 & \ddots & \vdots \\ & & & & t \\ & & & & 1 \end{bmatrix} \tag{9-88}
$$

（9）性质 9：若约当标准型矩阵 $\boldsymbol{J} = \mathrm{diag}[\boldsymbol{J}_1 \ \boldsymbol{J}_2 \ \cdots \ \boldsymbol{J}_l \ ]$，式中 $\boldsymbol{J}_i$ 为 $m_i \times m_i$ 阶约当块，则：

$$
e^{Jt} = \mathrm{diag}\left[e^{J_1 t} \ e^{J_2 t} \ \cdots \ e^{J_l t}\right] \tag{9-89}
$$

（10）性质 10：若 $n \times n$ 阶矩阵 $\boldsymbol{A}$ 有重特征根，$\boldsymbol{T}_J$ 是将 $\boldsymbol{A}$ 转化为约当标准型 $\boldsymbol{J}$ 的变换阵，即 $\boldsymbol{J} = \boldsymbol{T}_J^{-1}\boldsymbol{A}\boldsymbol{T}_J$，则：

$$
e^{At} = \boldsymbol{T}_J e^{Jt} \boldsymbol{T}_J^{-1} \tag{9-90}
$$

（11）性质 11：设 $\boldsymbol{A} = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix}$，$\boldsymbol{B} = \begin{bmatrix} \sigma & \omega \\ -\omega & \sigma \end{bmatrix}$，则 $e^{At} = \begin{bmatrix} \cos\omega t & \sin\omega t \\ -\sin\omega t & \cos\omega t \end{bmatrix}$，$e^{Bt} = e^{\sigma t} \cdot e^{At} = e^{\sigma t}\begin{bmatrix} \cos\omega t & \sin\omega t \\ -\sin\omega t & \cos\omega t \end{bmatrix}$

（12）性质 12：矩阵指数 e^{At} 可表示为有限项之和，即：

$$
e^{At} = \sum_{i=0}^{n-1} \boldsymbol{A}^i a_i(t) \tag{9-91}
$$

当 $\boldsymbol{A}$ 的 n 个特征根互不相等时，$a_i(t)$ 满足：

$$
\begin{bmatrix} a_0(t) \\ a_1(t) \\ \vdots \\ a_{n-1}(t) \end{bmatrix} = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^{n-1} \\ 1 & \lambda_2 & \lambda_2^2 & \cdots & \lambda_2^{n-1} \\ \vdots & \vdots & \vdots & \cdots & \cdots \\ 1 & \lambda_n & \lambda_n^2 & \cdots & \lambda_n^{n-1} \end{bmatrix}^{-1} \begin{bmatrix} e^{\lambda_1 t} \\ e^{\lambda_2 t} \\ \vdots \\ e^{\lambda_n t} \end{bmatrix} \tag{9-92}
$$

即满足：

$$
a_0(t) + a_1(t)\lambda_i + \cdots + a_{n-1}(t)\lambda_i^{\ n-1} = e^{\lambda_i t},\ (i = 1, 2, \cdots, n) \tag{9-93}
$$

若 $\boldsymbol{A}$ 有 n 重特征根，设 λ_1 为 m_1 重根，此时式（9-93）只有 $n - m_1 + 1$ 个独立方程，剩下的 $m_1 - 1$ 个方程可由下列关系添加：

$$
\frac{d^k}{d\lambda^k}\left\{a_0(t) + a_1(t)\lambda_i + \cdots + a_{n-1}(t)\lambda_1^{n-1} - e^{\lambda t}\right\}\Bigg|_{\lambda=\lambda_1} = 0,\ (k = 1, 2, \cdots, m_1 - 1) \tag{9-94}
$$

（13）性质 12：矩阵指数函数可用拉氏反变换法求得：

$$
e^{At} = L^{-1}\left\{[s\boldsymbol{I} - \boldsymbol{A}]^{-1}\right\} \tag{9-95}
$$

MATLAB 中提供了函数 expm( )来计算给定时刻的矩阵指数。

【例 9-8】　考虑如下矩阵 A：

$A=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 1 & -3 & 3\end{bmatrix}$，当 t=0.2 s 时，试求状态转移矩阵，即矩阵指数。

解：MATLAB 程序代码如下。

```
A=[0, 1, 0;0, 0, 1;1, -3, 3]
%状态转移时刻
t=0.2
%计算状态转移矩阵
Phi=expm(A*t)
```

运行结果如下：

```
Phi=
    1.0016    0.1954    0.0244
    0.0244    0.9283    0.2687
    0.2687   -0.7817    1.7344
```

由运算结果可知，该状态转移矩阵为：

$$\mathbf{Phi}=\begin{bmatrix}1.0016 & 0.1954 & 0.0244\\ 0.0244 & 0.9283 & 0.2687\\ 0.2687 & -0.7817 & 1.7344\end{bmatrix}$$

9.2.4.3　线性连续定常非齐次状态方程求解

线性连续定常非齐次状态方程为：

$$\dot{\boldsymbol{X}}=\boldsymbol{AX}+\boldsymbol{BU},\ \boldsymbol{X}\big|_{t=t_0}=\boldsymbol{X}(t_0) \tag{9-96}$$

从物理意义上看，系统从 t_0 时刻的初始状态 $X(t_0)$ 开始，在外界控制 $u(t)$ 的作用下运动。欲求系统在任意时刻的状态 $\boldsymbol{X}(t)$，就必须求解式（9-96）。

采用类似于齐次标量定常微分方程的解法，式（9-96）可写为：

$$\dot{\boldsymbol{X}}-\boldsymbol{AX}=\boldsymbol{BU} \tag{9-97}$$

两边同时左乘 e^{-At}，得：

$$\mathrm{e}^{-At}(\dot{\boldsymbol{X}}-\boldsymbol{AX})=\mathrm{e}^{-At}\boldsymbol{BU} \tag{9-98}$$

利用矩阵微积分知识，可由式（9-98）进一步得：

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathrm{e}^{-At}\boldsymbol{X})=\mathrm{e}^{-At}\boldsymbol{BU} \tag{9-99}$$

两边同时在 $[t_0,\ t]$ 区间积分，得：

$$\begin{aligned}&\mathrm{e}^{-At}X(t)\Big|_{t_0}^{t}=\int_{t_0}^{t}\mathrm{e}^{-A\tau}\boldsymbol{BU}(\tau)\mathrm{d}\tau\\ &\mathrm{e}^{-At}\boldsymbol{X}(t)-\mathrm{e}^{-At_0}\boldsymbol{X}(t_0)=\int_{t_0}^{t}\mathrm{e}^{-A\tau}\boldsymbol{BU}(\tau)\mathrm{d}\tau\end{aligned} \tag{9-100}$$

两边同时左乘 e^{At} 得：

$$X(t)=\mathrm{e}^{A(t-t_0)}X(t_0)+\mathrm{e}^{At}\int_{t_0}^{t}\mathrm{e}^{-A\tau}BU(\tau)\mathrm{d}\tau \tag{9-101}$$

即：

$$X(t)=\Phi(t-t_0)X(t_0)+\int_{t_0}^{t}\Phi(t-\tau)BU(\tau)\mathrm{d}\tau \tag{9-102}$$

当初始时刻 $t_0=0$ 时，式（9-101）变为：

$$X(t)=\Phi(t)X(0)+\int_{0}^{t}\Phi(t-\tau)BU(\tau)\mathrm{d}\tau \tag{9-103}$$

从式（9-102）和式（9-103）可知，非齐次状态方程式（9-96）的解由两部分组成，第一部分是在初始状态 $X(t_0)$ 作用下的自由运动，第二部分是在系统输入 $U(t)$ 作用下的强制运动。

下面讨论 $U(t)$ 为几种典型的控制输入时方程的解。

（1）脉冲信号输入

此时 $U(t)=K\delta(t)$，因此：

$$\begin{aligned}X(t)&=\mathrm{e}^{At}X(0)+\int_{0}^{t}\Phi(t-\tau)BK\delta(\tau)\mathrm{d}\tau\\&=\mathrm{e}^{At}X(0)+\mathrm{e}^{At}\left[\int_{0}^{t}\mathrm{e}^{-A\tau}\delta(\tau)\mathrm{d}\tau\right]BK=\mathrm{e}^{At}X(0)+\mathrm{e}^{At}BK\end{aligned} \tag{9-104}$$

即

$$X(t)=\mathrm{e}^{At}(X(0)+BK) \tag{9-105}$$

（2）阶跃信号输入

此时 $U(t)=K1(t)$，因此：

$$\begin{aligned}X(t)&=\mathrm{e}^{At}X(0)+\int_{0}^{t}\Phi(t-\tau)BK1(\tau)\mathrm{d}\tau\\&=\mathrm{e}^{At}X(0)+\mathrm{e}^{At}\left[\int_{0}^{t}\mathrm{e}^{-A\tau}\mathrm{d}\tau\right]BK=\mathrm{e}^{At}X(0)+\mathrm{e}^{At}\left[I-\mathrm{e}^{-At}\right]A^{-1}BK\end{aligned} \tag{9-106}$$

即：

$$X(t)=\mathrm{e}^{At}X(0)+\left[\mathrm{e}^{At}-I\right]A^{-1}BK \tag{9-107}$$

（3）斜坡信号输入

此时 $U(t)=Kt$，因此：

$$X(t)=\mathrm{e}^{At}X(0)+\left[A^{-2}(\mathrm{e}^{At}-I)-A^{-1}t\right]BK \tag{9-108}$$

9.2.4.4　离散时间系统状态方程求解

数字计算机处理的是时间上离散的数字量，如果要采用数字计算机对连续时间系统进行控制，就必须将连续系统状态方程离散化。因此，在对离散时间状态方程进行求解前，必须将连续状态空间表达式离散化。

设连续系统动态方程为：

$$\begin{cases}\dot{X}=AX+BU\\Y=CX+DU\end{cases} \tag{9-109}$$

系统离散化的原则是：在每个采样时刻 $kT(k=0,1,2\cdots)$ ，T 为采样周期，系统离散化前后的 $\boldsymbol{U}(kt), \boldsymbol{X}(kt), \boldsymbol{Y}(kt)$ 保持不变。而采样的方法是在 $t=kT$ 时刻对 $\boldsymbol{U}(t)$ 值采样，得 $\boldsymbol{U}(kT)$ ，并通过零阶保持器，使 $\boldsymbol{U}(kt)$ 的值在 $[kT,\ (k+1)T]$ 时间段保持不变。

根据上述离散化原则，得到离散化后的动态方程为：

$$\begin{cases}\boldsymbol{X}\left[(k+1)T\right]=\boldsymbol{G}(T)\boldsymbol{X}(kT)+\boldsymbol{H}(T)\boldsymbol{U}(kT)\\ \boldsymbol{Y}(kT)=\boldsymbol{C}\boldsymbol{X}(kT)+\boldsymbol{D}\boldsymbol{U}(kT)\end{cases} \tag{9-110}$$

上述输出方程表示了 kT 时刻离散系统的输出 $\boldsymbol{Y}(kT)$ 和输入 $\boldsymbol{U}(kT)$ 及其系统状态量 $\boldsymbol{X}(kT)$ 之间的关系，它应该与离散化前的关系一样。

根据连续时间状态方程求解公式，假设 $t_0=kT$ ，求 $t=(k+1)T$ 时刻的状态 $\boldsymbol{X}[(k+1)T]$ 。注意到 $\boldsymbol{U}(t)=\boldsymbol{U}(kT)$ 在 $kT\sim(k+1)T$ 时段保持不变，因此 $\boldsymbol{X}[(k+1)T]$ 为：

$$\begin{aligned}\boldsymbol{X}\left[(k+1)T\right]&=\mathrm{e}^{\boldsymbol{A}t}\boldsymbol{X}(kT)+\int_{kT}^{(k+1)T}\mathrm{e}^{\boldsymbol{A}[(k+1)T-\tau]}\boldsymbol{B}\boldsymbol{U}(kT)\mathrm{d}\tau\\ &=\mathrm{e}^{\boldsymbol{A}t}\boldsymbol{X}(kT)+\int_{kT}^{(k+1)T}\mathrm{e}^{\boldsymbol{A}[(k+1)T-\tau]}\boldsymbol{B}\cdot\mathrm{d}\tau\cdot\boldsymbol{U}(kT)\\ &=\boldsymbol{G}(T)\boldsymbol{X}(kT)+\boldsymbol{H}(T)\boldsymbol{U}(kT)\end{aligned} \tag{9-111}$$

其中 $\boldsymbol{G}(T)=\mathrm{e}^{\boldsymbol{A}T}$ ，它只与采样周期 $\boldsymbol{T}$ 有关，$\boldsymbol{H}(T)=\int_{kT}^{(k+1)T}\mathrm{e}^{\boldsymbol{A}[(k+1)T-\tau]}\boldsymbol{B}\cdot\mathrm{d}\tau$ 。

令 $t=(k+1)T-\tau$ ，则 $\mathrm{d}t=-\mathrm{d}\tau$ ，$\begin{cases}\tau=kT\text{时}, & t=T\\ \tau=(k+1)T\text{时}, & t=0\end{cases}$

因此：

$$\boldsymbol{H}(T)=\int_{\tau}^{0}\mathrm{e}^{\boldsymbol{A}t}\boldsymbol{B}(-\mathrm{d}t)=\int_{0}^{\tau}\mathrm{e}^{\boldsymbol{A}t}\boldsymbol{B}\mathrm{d}t=\boldsymbol{A}^{-1}(\mathrm{e}^{\boldsymbol{A}\tau}-\boldsymbol{I})\boldsymbol{B} \tag{9-112}$$

式（9-112）也只与采样周期 $\boldsymbol{T}$ 有关。为了方便起见，在书写时通常忽略 kT 时刻的 $\boldsymbol{T}$ 符号，直接用 k 代表 kT 时刻。因此连续系统离散化公式为：

$$\begin{cases}\boldsymbol{X}(k+1)=\boldsymbol{G}(T)\boldsymbol{X}(k)+\boldsymbol{H}(T)\boldsymbol{U}(k)\\ \boldsymbol{Y}(k)=\boldsymbol{C}\boldsymbol{X}(k)+\boldsymbol{D}\boldsymbol{U}(k)\end{cases} \tag{9-113}$$

其中，

$$\boldsymbol{G}(T)=\mathrm{e}^{\boldsymbol{A}T},\boldsymbol{H}(T)=\int_{0}^{\tau}\mathrm{e}^{\boldsymbol{A}t}\boldsymbol{B}\mathrm{d}t=\boldsymbol{A}^{-1}(\mathrm{e}^{\boldsymbol{A}\tau}-\boldsymbol{I})\boldsymbol{B} \tag{9-114}$$

离散时间状态方程求解一般有两种方法：递推法（迭代法）和 Z 变换法。前者对定常、时变系统都适用，而后者只适用于定常系统。此处只介绍递推法，Z 变换法将在后面章节介绍。

对于线性定常离散系统状态方程：

$$\boldsymbol{X}(k+1)=\boldsymbol{G}(T)\boldsymbol{X}(k)+\boldsymbol{H}(T)\boldsymbol{U}(k),\ \boldsymbol{X}(k)\big|_{k=0}=\boldsymbol{X}(0) \tag{9-115}$$

依次取 $k=0,1,2,\cdots$ ，得：

$$\boldsymbol{X}(1)=\boldsymbol{G}\boldsymbol{X}(0)+\boldsymbol{H}\boldsymbol{U}(0)$$

$$\boldsymbol{X}(2)=\boldsymbol{G}\boldsymbol{X}(1)+\boldsymbol{H}\boldsymbol{U}(1)=\boldsymbol{G}^2\boldsymbol{X}(0)+\boldsymbol{G}\boldsymbol{H}\boldsymbol{U}(0)+\boldsymbol{H}\boldsymbol{U}(1)$$

$$\boldsymbol{X}(3)=\boldsymbol{G}\boldsymbol{X}(2)+\boldsymbol{H}\boldsymbol{U}(2)=\boldsymbol{G}^3\boldsymbol{X}(0)+\boldsymbol{G}^2\boldsymbol{H}\boldsymbol{U}(0)+\boldsymbol{G}\boldsymbol{H}\boldsymbol{U}(1)+\boldsymbol{H}\boldsymbol{U}(2)$$

$$\begin{aligned}
\boldsymbol{X}(k)&=\boldsymbol{G}\boldsymbol{X}(k-1)+\boldsymbol{H}\boldsymbol{U}(k-1)\\
&=\boldsymbol{G}^k\boldsymbol{X}(0)+\boldsymbol{G}^{k-1}\boldsymbol{H}\boldsymbol{U}(0)+\cdots+\boldsymbol{G}\boldsymbol{H}\boldsymbol{U}(k-2)+\boldsymbol{H}\boldsymbol{U}(k-1)\\
&=\boldsymbol{G}^k\boldsymbol{X}(0)+\sum_{j=0}^{k-1}\boldsymbol{G}^j\boldsymbol{H}\boldsymbol{U}(k-1-j)\\
&=\boldsymbol{G}^k\boldsymbol{X}(0)+\sum_{j=0}^{k-1}\boldsymbol{G}^{k-1-j}\boldsymbol{H}\boldsymbol{U}(j)
\end{aligned}\tag{9-116}$$

当初始时刻为 h 时，同理可推出：

$$\boldsymbol{X}(k)=\boldsymbol{G}^{k-h}\boldsymbol{X}(h)+\sum_{j=h}^{k-1}\boldsymbol{G}^{k-1-j}\boldsymbol{H}\boldsymbol{U}(j)\tag{9-117}$$

与连续时间系统方程解类似，记为 $\boldsymbol{\Phi}(k)=\boldsymbol{G}^k$ 或 $\boldsymbol{\Phi}(k-h)=\boldsymbol{G}^{k-h}$，称它们为离散系统的状态转移矩阵。因此离散系统的解可记为：

$$\boldsymbol{X}(k)=\boldsymbol{\Phi}(k)\boldsymbol{X}(0)+\sum_{j=0}^{k-1}\boldsymbol{\Phi}(k-1-j)\boldsymbol{H}\boldsymbol{U}(j)\tag{9-118}$$

或者

$$\boldsymbol{X}(k)=\Phi(k-h)\boldsymbol{X}(h)+\sum_{j=h}^{k-1}\Phi(k-1-j)\boldsymbol{H}\boldsymbol{U}(j)\tag{9-119}$$

9.2.4.5　采用 MATLAB 求解状态方程

【例 9-9】　已知系统的状态空间模型为 $\boldsymbol{A}=\begin{bmatrix}-2 & -2.5 & -0.5\\ 1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$，$\boldsymbol{B}=\begin{bmatrix}1\\0\\0\end{bmatrix}$，$\boldsymbol{C}=[0\quad 1.5\quad 1]$，$\boldsymbol{D}=0$，试求：

（1）系统的脉冲响应和阶跃响应；

（2）在初始状态为 $x(0)=[1\quad 0\quad 2]^{\mathrm{T}}$ 的条件下，输入 $u(t)=\begin{cases}2 & 0\leqslant t<2\\ 0.5 & t\geqslant 2\end{cases}$ 时，状态变量 $\boldsymbol{X}(t)=[x_1(t),\quad x_2(t),\quad x_3(t)]^{\mathrm{T}}$ 的响应曲线。

解：MATLAB 程序代码如下。

```
A=[-2 -2.5 -0.5; 1 0 0; 0 1 0]
B=[1; 0; 0]
C=[0 1.5 1]
D=0
G=ss(A, B, C, D)
t=[0:0.1:20]';
% 计算脉冲响应
impulse(G, t)
grid
% 计算阶跃响应
```

```
step(G, t)
grid
```

输出结果如图 9.9 和图 9.10 所示。

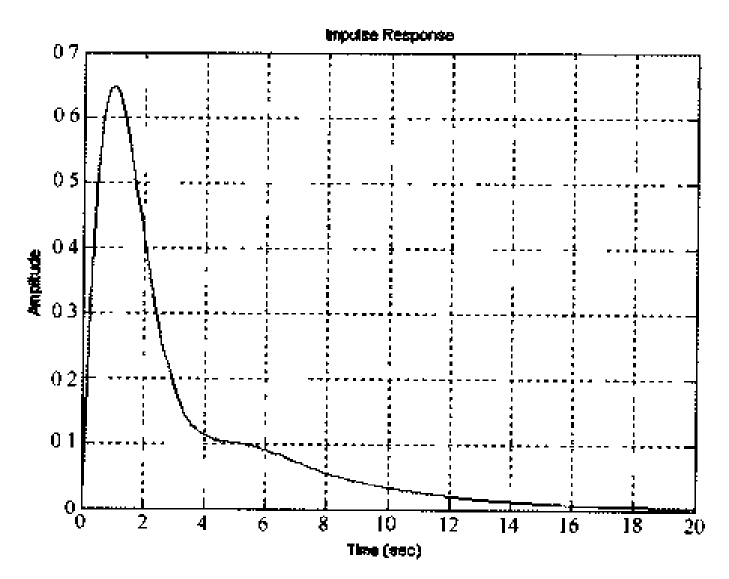


图 9.9　例 9-9 系统脉冲响应曲线

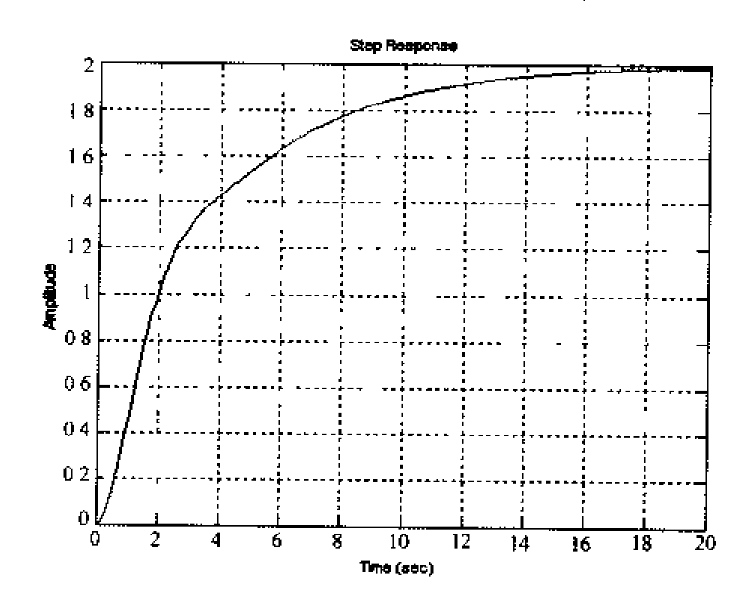


图 9.10　例 9-9 系统阶跃响应曲线

接着输入如下 MATLAB 程序代码：

```
% 初始状态
xo=[1; 0; 2]
  % 输入量
u(1, 1:20)=2*ones(1, 20);
  u(1, 21:201)=0.5*ones(1, 181);
  % 计算输入响应
```

```
[y, t, x]=lsim(G, u, t, xo);
  %绘制曲线
plot(t, x(:, 1), '-', t, x(:, 2), '--', t, x(:, 3), '-.')
  xlabel('时间/秒 ')
  ylabel('幅值')
  grid
  text(6, 0.3, 'x_1(t)')
  text(6, -0.5, 'x_2(t)')
  text(8, 1.8, 'x_3(t)')
```

输出结果如图 9.11 所示。

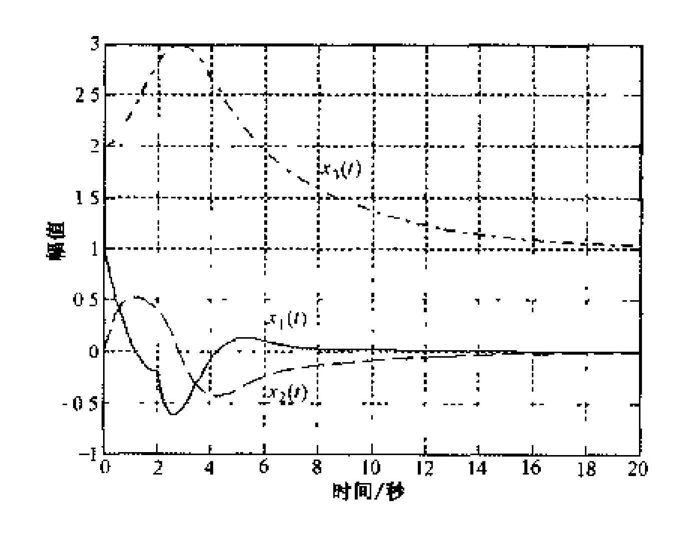


图 9.11　例 9-9 系统状态变化曲线

9.3　线性系统的状态可控性与状态可观性

在现代控制理论中，可控性（Controllability）和可观性（Observability）是两个重要的概念，它们是卡尔曼（Kalman）在 1960 年提出的，是最优控制和最优估计的设计基础。

可观（测）性针对的是系统状态空间模型中状态的可观测性，它表示系统内部状态（通常是不可以直接测量的）可由系统输出量 $Y(t)$（通常是可以直接测量的）反映的能力。

严格地说，可控性分为有两种，一种是系统控制输入 $U(t)$ 对系统内部状态 $X(t)$ 的控制能力，另一种是控制输入 $U(t)$ 对系统输出 $Y(t)$ 的控制能力。但一般没有特别指明时，指的都是状态的可控性。

系统的可控性和可观性研究一般都基于系统的状态空间表达式。

9.3.1　状态可控性

9.3.1.1　可控的定义

设单输入 n 阶线性定常离散系统状态方程为：

$$X(k+1)=GX(k)+HU(k) \tag{9-120}$$

其中 $X(k)$ 为 n 维状态向量；$U(k)$ 为 1 维输入向量；G 为 $n\times n$ 系统矩阵；H 为 $n\times 1$ 输入矩阵。如果存在有限步的控制信号序列 $U(k)$，$U(k+1)$，…，$U(N-1)$，使得系统第 k 步的状态 $X(k)$ 能在第 N 步到达零状态，即 $X(N)=0$，其中 N 是大于 k 的有限正整数，那么就说系统在第 k 步的状态 $X(k)$ 是可控的；如果第 k 步的所有状态都可控，则称式（9-120）所表示的系统在第 k 步是完全可控的；进一步地，如果系统的每一步都是可控的，则称式（9-120）所表示的系统是完全可控的，或称系统为可控系统。

设单输入 n 阶线性定常连续系统为：

$$\dot{X}=AX+BU \tag{9-121}$$

若存在一个分段连续的控制函数 $U(t)$，能在有限的时间段 $[t_0,t_f]$ 内把将系统从 t_0 时刻的初始状态 $X(t_0)$ 转移至任意指定的终态 $X(t_f)$，则称式（9-121）所表示的系统在 t_0 时刻的状态 $X(t_0)$ 是可控的；如果系统每一个状态 $X(t_0)$ 都可控，则称系统是状态完全可控的；反之，只要有一个状态不可控，就称系统是不可控的。

对于线性定常连续系统，为简便起见，可假设 $t_0=0$，$X(t_f)=0$，即 0 时刻的任意初始状态 $X(0)$ 在有限时间段内转移至零状态（原点）。

9.3.1.2　可控的判据

单输入 n 阶离散系统可控的充分必要条件是：可控判别阵 $M=[h \quad Gh \quad \cdots \quad G^{n-1}h]_{n\times n}$ 的秩等于 n，即

$$\text{rank}(M)=\text{rank}[h \quad Gh \quad \cdots \quad G^{n-1}h]=n \tag{9-122}$$

n 阶连续系统可控的充分必要条件为可控判别阵 $M=[b \quad Ab \quad \cdots \quad A^{n-1}b]_{n\times n}$ 的秩等于 n。

对于多输入 n 阶连续定常系统：

$$\dot{X}=AX+BU \tag{9-123}$$

其中，A 为 $n\times n$ 阶阵，B 为 $n\times r$ 阶阵，U 为 r 维输入。系统可控的充分必要条件为可控判别阵 $M=\begin{bmatrix}B & AB & \cdots & A^{n-1}B\end{bmatrix}_{n\times n}$ 的秩等于 n，即 $\text{rank}(M)=n$。

对于式（9-123）所表示的系统状态方程，如果输入 $U(t)$ 对状态 $X(t)$ 的传递函数（阵）没有零极点对消，那么系统可控，否则系统不可控。

对连续系统：

$$\begin{cases}\dot{X}=AX+BU\\ Y=CX+DU\end{cases} \tag{9-124}$$

其中，X 为 n 维状态向量，Y 为 m 维输出向量，U 为 r 维控制向量，A 为 $n\times m$ 阶阵，B 为 $n\times r$ 阶阵，C 为 $m\times n$ 阶阵，D 为 $m\times r$ 阶阵。如果 $m\times(n+1)r$ 阶矩阵 $[CB \quad CAB \quad CA^2B \quad \cdots \quad CA^{n-1}B \quad D]$ 的秩为 m，那么系统是输出可控的。也就是说对任意给定输出初始量 $Y(t_0)$，总能找到一个分段连续的控制 $U(t)$，使系统输出能在有限的时间 $[t_0,\ t_f]$ 段内，转移至任一指定的输出 $Y(t_f)$。

MATLAB 提供了求系统可控判别矩阵 $M=[B, AB, A^2B\ \ldots]$ 的函数 ctrb(A, B)，以及求 M 秩的函数 rank（M）。

【例 9-10】　已知系统的矩阵 $A=\begin{bmatrix}0 & 1\\ 2.5 & -1.5\end{bmatrix}$，$B=\begin{bmatrix}1\\1\end{bmatrix}$，试判断系统是否可控。

解：MATLAB 程序代码如下。

```
A=[0, 1;2.5, -1.5];
B=[1;1];
%计算可控判别矩阵
M=ctrb(A, B)
%计算可控判别矩阵的秩
R=rank(M)
```

运行结果如下：

```
M=
      1     1
      1     1
R=
      1
```

由计算结果可知，$R<n$，系统是不可控的。

9.3.2　状态可观性

9.3.2.1　可观的定义

对于线性定常离散系统：

$$\begin{cases}X(k+1)=GX(k)+HU(k)\\ Y(kT)=CX(k)\end{cases} \tag{9-125}$$

其中，$X(k)$ 为 n 维状态向量；$U(k)$ 为 1 维输入向量；$Y(k)$ 为 1 维输出向量；G 为 $n\times n$ 系统矩阵；H 为 $n\times 1$ 输入矩阵；C 为 $1\times n$ 输出矩阵。如果根据第 i 步以及之后有限步的输出观测 $y(i), y(i+1),\cdots, y(N)$，就能惟一确定第 i 步的状态 $X(i)$，则称系统式（9-125）所表示的系统是可观的。

对于线性定常离散系统，不失一般性，可设 $i=0$，即从第 0 步开始观测，确定的是 $X(0)$ 的值，并且由于 $U(k)$ 不影响系统的可观性，因此可令 $U(k)\equiv 0$，则式（9-125）变为：

$$\begin{cases}X(k+1)=GX(k)\\ Y(kT)=CX(k)\end{cases} \tag{9-126}$$

对于线性连续系统方程：

$$\begin{cases}\dot{X}=AX+BU\\ Y=CX\end{cases} \tag{9-127}$$

若对任意给定的输入 $U(t)$，总能在有限的时间段 $[t_0, t_f]$ 内，根据系统的输入 $U(t)$ 及系统观测 $Y(t)$，能惟一地确定 t_0 时刻的每一状态 $X(t_0)$，那么称系统在 t_0 时刻是状态可观测的。

若系统在所讨论时间段内每一时刻都可观测，则称系统是完全可观测的。

9.3.2.2 可观的判据

对于式（9-125）所表示的离散系统，其完全可观的充分必要条件为可观判别阵 $N=\begin{bmatrix} C \\ CG \\ \vdots \\ CG^{n-1} \end{bmatrix}$ 的秩等于 n，即 $\text{rank}(N)=n$。

线性连续系统完全可观的充分必要条件是可观判别阵 $N=\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$ 的秩为 n。

MATLAB 提供了求系统可观判别矩阵 N=[*C*; CA; CA^2 ...]的函数 obsv(A, C)，以及求 N 秩的函数 rank（N）。

【例 9-11】 已知系统的矩阵 $A=\begin{bmatrix} 2 & 3 \\ -1 & -2 \end{bmatrix}$，$C=\begin{bmatrix} 2 & 0 \\ -1 & 1 \end{bmatrix}$，试判断系统是否可观。

解：MATLAB 程序代码如下。

```
A=[2, 3; -1, -2];
C=[2, 0; -1, 1];
%计算可观判别矩阵
N=obsv(A, C)
%计算可观判别矩阵的秩
R=rank(N)
```

运行结果如下：

```
N=
     2     0
    -1     1
     4     6
    -3    -5
R=
     2
```

由计算结果可知，$R=n=2$，系统是可观的。

9.3.3 对偶系统和对偶原理

9.3.3.1 对偶系统

设系统 $\sum 1$ 的动态方程为：

$$\begin{cases}\dot{X}_1 = A_1X_1 + B_1U_1 \\ Y_1 = C_1X_1\end{cases} \tag{9-128}$$

系统$\sum 2$的动态方程为：

$$\begin{cases}\dot{X}_2 = A_2X_2 + B_2U_2 \\ Y_2 = C_2X_2\end{cases} \tag{9-129}$$

若$\sum 1$，$\sum 2$满足：$A_2 = A_1^{\mathrm{T}}$，$B_2 = C_1^{\mathrm{T}}$，$C_2 = B_1^{\mathrm{T}}$，则称$\sum 1$和$\sum 2$互为对偶系统。

对偶系统具有以下基本性质：

（1）如果将$\sum 1$模拟结构图中信号线反向，即输入端变输出端，输出端变输入端，信号综合点变信号引出点，信号引出点变信号综合点，那么形成的就是$\sum 2$的模拟结构图，如图 9.12 所示。

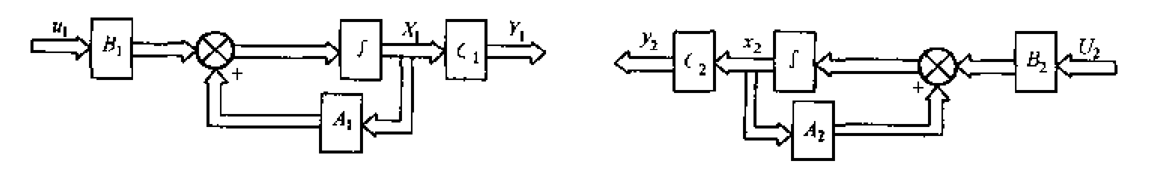


图 9.12　对偶系统结构图

（2）对偶系统的传递函数阵互为转置。

$$\begin{aligned}W_2(s) &= C_2(sI - A_2)^{-1}B_2 = B_1^{\mathrm{T}}(sI - A_1^{T})^{-1}C_1^{\mathrm{T}} \\ &= B_1^{\mathrm{T}}\left[(sI - A_1^{\mathrm{T}})^{-1}\right]^{\mathrm{T}}C_1^{\mathrm{T}} = \left[(sI - A_1^{\mathrm{T}})^{-1}B_1\right]^{\mathrm{T}} = \left[W_1(s)\right]^{\mathrm{T}}\end{aligned} \tag{9-130}$$

因此若$\sum 1$，$\sum 2$为单入单出（SISO）系统，则有$W_1(s) = W_2(s)$。

（3）对偶系统特征方程式相同。

$|sI - A_2| = |sI - A_1^{\mathrm{T}}| = (|sI - A_1|)^{\mathrm{T}} = 0$，即$|sI - A_2| = 0$和$|sI - A_1| = 0$是等价的。

9.3.3.2　对偶原理

若系统$\sum 1 = (A_1, B_1, C_1)$和$\sum 2 = (A_2, B_2, C_2)$互为对偶系统，则$\sum 1$的可控性等价于$\sum 2$的可观性，$\sum 1$的可观性等价于$\sum 2$的可控性，即：

$$N_2 = \begin{bmatrix} C_2 \\ C_2A_2 \\ \vdots \\ C_2A_2^{n-1}\end{bmatrix} = \begin{bmatrix} B_1^{\mathrm{T}} \\ B_1^{\mathrm{T}}A_1^{\mathrm{T}} \\ \vdots \\ B_1^{\mathrm{T}}(A_1^{\mathrm{T}})^{n-1}\end{bmatrix} = \begin{bmatrix} B_1^{\mathrm{T}} \\ (A_1B_1)^{\mathrm{T}} \\ \vdots \\ (A_1^{n-1}B_1)^{\mathrm{T}}\end{bmatrix} = \begin{bmatrix} B_1 & A_1B_1 & \cdots & A_1^{n-1}B_1\end{bmatrix}^{\mathrm{T}} = M_1^{\mathrm{T}} \tag{9-131}$$

因此$\mathrm{rank}(N_2) = \mathrm{rank}(M_1)$，即$\sum 1$的可控性等价于$\sum 2$的可观性。

9.3.4　可控标准型和可观标准型

9.3.4.1　可控标准型

控制系统的可控标准型有两种形式，分别称之为可控Ⅰ型和可控Ⅱ型。

对于可控 I 型 $\sum c1(\boldsymbol{A}_{c1},\boldsymbol{B}_{c1},\boldsymbol{C}_{c1})$，其各矩阵的形式为：

$$\begin{cases}\boldsymbol{A}_{c1}=\begin{bmatrix}0 & 1 & 0 & \cdots & 0\\ 0 & 0 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & \cdots & 1\\ -a_0 & -a_1 & \cdots & \cdots & -a_{n-1}\end{bmatrix},\boldsymbol{B}_{c1}=\begin{bmatrix}0\\ \vdots\\ \vdots\\ 0\\ 1\end{bmatrix}\\ \boldsymbol{C}_{c1}=[\beta_0 \quad \beta_1 \quad \cdots \quad \beta_{n-1}]\end{cases} \tag{9-132}$$

对于可控 II 型 $\sum c2(\boldsymbol{A}_{c2},\boldsymbol{B}_{c2},\boldsymbol{C}_{c2})$，其各矩阵的形式为：

$$\begin{cases}\boldsymbol{A}_{c1}=\begin{bmatrix}0 & \cdots & \cdots & \cdots & 0 & -a_0\\ 1 & 0 & \cdots & \cdots & 0 & -a_1\\ 0 & 1 & 0 & \cdots & \cdots & \vdots\\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots\\ 0 & \cdots & \cdots & \cdots & 1 & -a_{n-1}\end{bmatrix},\boldsymbol{B}_{c1}=\begin{bmatrix}1\\ 0\\ \vdots\\ \vdots\\ 0\end{bmatrix}\\ \boldsymbol{C}_{c1}=[\beta_0 \quad \beta_1 \quad \cdots \quad \beta_{n-1}]\end{cases} \tag{9-133}$$

注意：$\boldsymbol{C}_{c1}$ 中的 β_i 与 $\boldsymbol{C}_{c2}$ 中的 β_i 不是同一数值。

$\sum c1$ 的模拟结构图如图 9.13 所示，$\sum c2$ 结构图如图 9.14 所示。

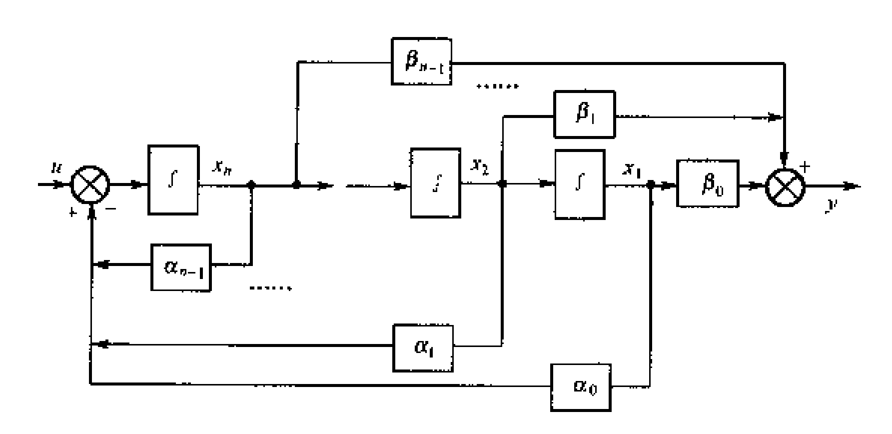


图 9.13　可控 I 型模拟结构图

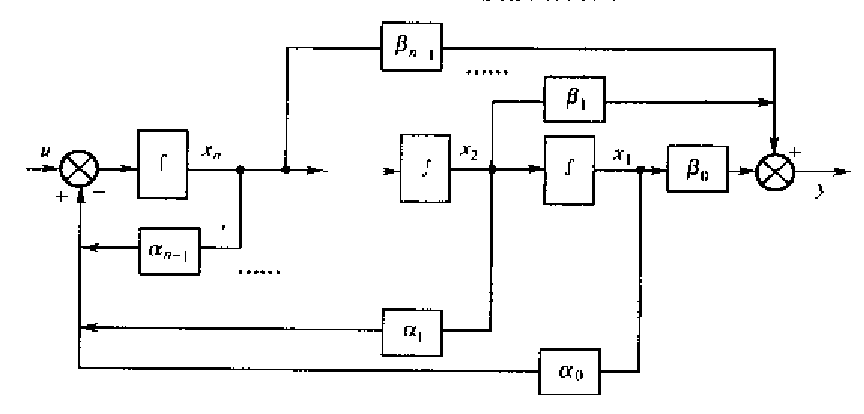


图 9.14　可控 II 型模拟结构图

$\sum c1(A_{c1},B_{c1},C_{c1})$ 和 $\sum c2(A_{c2},B_{c2},C_{c2})$ 之所以称为可控型，主要是这种形式的动态方程所表示的系统是可控的。

动态方程到可控标准型的转化步骤如下所述。

对于一般动态方程：

$$\begin{cases}\dot{X} = AX + BU \\ Y = CX\end{cases} \tag{9-134}$$

如果系统可控，即 $\text{rank}[b \quad Ab \quad \cdots \quad A^{n-1}b]$ 满秩，那么可以通过非奇异矩阵

$$T_{c1} = \begin{bmatrix}A^{n-1}b & A^{n-2}b & \cdots & Ab & b\end{bmatrix}\begin{bmatrix}1 & \cdots & \cdots & \cdots & 0 \\ a_{n-1} & 1 & \cdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ a_2 & \cdots & \cdots & \cdots & \vdots \\ a_1 & a_2 & \cdots & a_{n-1} & 1\end{bmatrix} \tag{9-135}$$

的线性变换，将系统 $\sum(A,B,C)$ 变换成可控 I 型 $\sum c1(A_{c1},B_{c1}C_{c1})$。

$$\begin{cases}A_{c1} = T_{c1}^{-1}AT_{c1} \\ B_{c1} = T_{c1}^{-1}B \\ C_{c1} = CT_{c1}\end{cases} \tag{9-136}$$

其中 $a_i\ (i = 0,1,2,\cdots,n-1)$ 是系统特征多项式的系数，即：

$$|\lambda I - A| = \lambda^n + a_{n-1}\lambda + \cdots + a_1\lambda + a_0 \tag{9-137}$$

由可控 I 型求系统传递函数是非常方便的，因为：

$$\begin{aligned} W(s) &= C_{c1}(sI - A_{c1})^{-1}B_{c1} \\ &= (\beta_0 \quad \beta_1 \quad \cdots \quad \beta_{n-1})\begin{bmatrix}s & -1 & \cdots & \cdots & 0 \\ 0 & s & -1 & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & \vdots & \cdots & \vdots & -1 \\ a_0 & a_1 & \cdots & a_{n-1} & s + a_{n-1}\end{bmatrix}^{-1}\begin{bmatrix}0 \\ \vdots \\ \vdots \\ 0 \\ 1\end{bmatrix} \\ &= \frac{\beta_{n-1}s^{n-1} + \beta_{n-2}s^{n-2} + \cdots + \beta_1 s + \beta_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}\end{aligned} \tag{9-138}$$

由式（9-138）可知，根据系统的传递函数 $W(s)$ 可直接写出系统的可控 I 型，反之亦然。

对于式（9-134）所示的一般动态系统，如果系统可控，即 $\text{rank}\begin{bmatrix}b & Ab & \cdots & A^{n-1}b\end{bmatrix}$ 满秩，那么可以通过非奇异变换：

$$T_{c2} = \begin{bmatrix}b & Ab & \cdots & A^{n-1}b\end{bmatrix} \tag{9-139}$$

将系统 $\sum(A,B,C)$ 变换成可控 II 型 $\sum c2(A_{c2},B_{c2},C_{c2})$，其中：

$$\begin{cases}A_{c2} = T_{c2}^{-1}AT_{c2} \\ B_{c2} = T_{c2}^{-1}B \\ C_{c2} = CT_{c2} = \begin{bmatrix}CB & CAB & \cdots & CA^{n-1}B\end{bmatrix}\end{cases} \tag{9-140}$$

A_{c2} 中的 $a_i\ (i=0,1,2,\cdots,n-1)$ 是系统特征多项式的系数，即

$$|\lambda I - A| = \lambda^n + a_{n-1}\lambda + \cdots + a_1\lambda + a_0 \tag{9-141}$$

MATLAB 提供了函数 ss2ss(A, B, C, D, T)，用于状态空间模型（A, B, C, D）的线性变换。线性变换为 $Z = TX$，其中 X 为系统的状态变量。设原系统的状态空间描述为：

$$\begin{cases}\dot{X} = AX + BU \\ Y = CX + DU\end{cases}$$

则变换后的状态空间描述为：

$$\begin{cases}\dot{Z} = (TAT^{-1})Z + (TB)U \\ Y = (CT^{-1})Z + DU\end{cases}$$

【例 9-12】 已知系统的系数矩阵 $A=\begin{bmatrix}1 & 2 & 0\\ 3 & -1 & 1\\ 0 & 2 & 0\end{bmatrix}$，$B=\begin{bmatrix}2\\1\\1\end{bmatrix}$，$C=\begin{bmatrix}0 & 0 & 1\end{bmatrix}$，$D=0$，试判断它的可控性，如果完全可控，将其转化为可控 II 型。

解：MATLAB 程序代码如下。

```
A=[1, 2, 0;3, -1, 1;0, 2, 0];
B=[2;1;1];
C=[0, 0, 1]
D=0
T=ctrb(A, B)
R=rank(T)
```

运行结果如下：

```
T=
     2     4    16
     1     6     8
     1     2    12
R=
     3
```

由运算结果“R=3”可知，系统完全可控，可以将其转化为可控 II 型。

接着输入如下 MATLAB 程序代码：

```
[Ac2, Bc2, Cc2, Dc2]=ss2ss(A, B, C, D, inv(T))
```

运行结果如下：

```
Ac2=
     0     0    -2
     1     0     9
     0     1     0
```

```
Bc2=
     1
     0
     0
Cc2=
     1     2     12
Dc2=
     0
```

由计算结果可知，该系统的可控 II 型为 $\begin{cases}\dot{\boldsymbol{Z}}=\begin{bmatrix}0 & 0 & -2\\ 1 & 0 & 9\\ 0 & 1 & 0\end{bmatrix}\boldsymbol{Z}+\begin{bmatrix}1\\0\\0\end{bmatrix}\boldsymbol{U}\\ \boldsymbol{Y}=\begin{bmatrix}1 & 0 & 0\end{bmatrix}Z\end{cases}$。

9.3.4.2　可观标准型

控制系统的可观标准型也有两种形式，可观 I 型 $\sum o1(\boldsymbol{A}_{o1},\boldsymbol{B}_{o1},\boldsymbol{C}_{o1})$ 和可观 II 型 $\sum o2(\boldsymbol{A}_{o2},\boldsymbol{B}_{o2},\boldsymbol{C}_{o2})$，其中：

$$\begin{cases}\boldsymbol{A}_{o1}=\begin{bmatrix}0 & 1 & 0 & \cdots & 0\\ 0 & 0 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & \cdots & 1\\ -a_0 & -a_1 & \cdots & \cdots & -a_{n-1}\end{bmatrix},\boldsymbol{B}_{o1}=\begin{bmatrix}\beta_0\\ \beta_1\\ \vdots\\ \vdots\\ \beta_{n-1}\end{bmatrix}\\ \boldsymbol{C}_{o1}=\begin{bmatrix}1 & 0 & \cdots & & 0\end{bmatrix}\end{cases} \tag{9-142}$$

$$\begin{cases}\boldsymbol{A}_{o1}=\begin{bmatrix}0 & \cdots & \cdots & \cdots & 0 & -a_0\\ 1 & 0 & \cdots & \cdots & 0 & -a_1\\ 0 & 1 & 0 & \cdots & \cdots & \vdots\\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots\\ 0 & \cdots & \cdots & \cdots & 1 & -a_{n-1}\end{bmatrix},\boldsymbol{B}_{o1}=\begin{bmatrix}\beta_0\\ \beta_1\\ \vdots\\ \vdots\\ \beta_{n-1}\end{bmatrix}\\ \boldsymbol{C}_{o1}=\begin{bmatrix}0 & \cdots & 0 & 1 & & \end{bmatrix}\end{cases} \tag{9-143}$$

注意：$\sum o1$ 中的 β_i 与 $\sum o2$ 中的 β_i 不是同一数值。

可观 I 型 $\sum o1$ 的模拟结构图如图 9.15 所示，可观 II 型 $\sum o2$ 的模拟结构图如图 9.16 所示。

$\sum o1(\boldsymbol{A}_{o1},\boldsymbol{B}_{o1},\boldsymbol{C}_{o1})$ 和 $\sum o2(\boldsymbol{A}_{o2},\boldsymbol{B}_{o2},\boldsymbol{C}_{o2})$ 之所以称为可观标准型，主要是这种形式的动态方程所表示的系统是可观的。

动态方程到可观标准型的转化步骤如下所述。

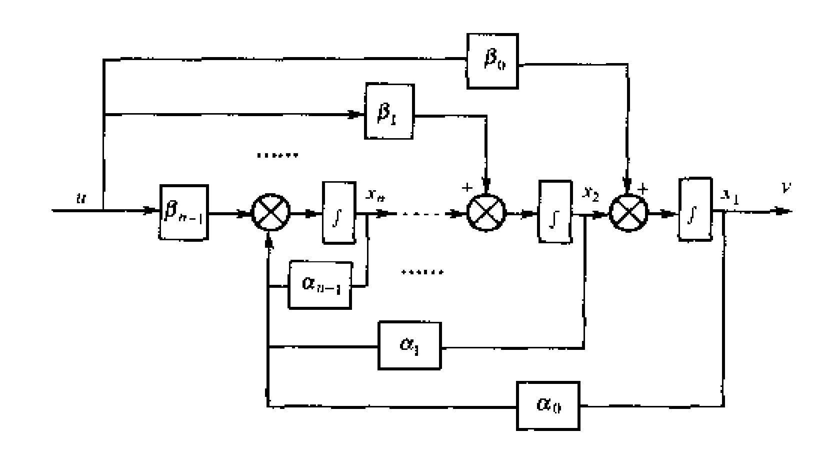


图 9.15　可观 I 型模拟结构图

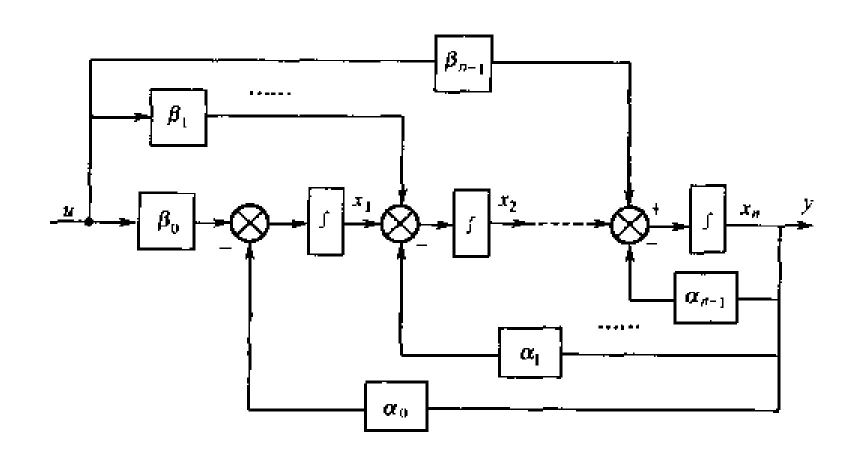


图 9.16　可观 II 型模拟结构图

对一般动态方程：

$$\begin{cases} \dot{X} = AX + BU \\ Y = CX \end{cases} \tag{9-144}$$

如果系统可观，即系统可观判别阵

$$N = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

满秩，那么可以通过

$$T_{o1} = N^{-1} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}^{-1} \tag{9-145}$$

的变换，将系统 $\sum(A,B,C)$ 变换成可观 I 型，其中 B_{o1} 中的 $\beta_i = CA_i b\,(i = 0,1,2,\cdots,n-1)$。

对式（9-144）所示的系统，如果系统可观，那么通过矩阵

$$T_{o2}=\left[\begin{bmatrix}1 & a_{n-1} & \cdots & \cdots & a_1\\ 0 & 1 & a_{n-1} & \ddots & a_2\\ 0 & 0 & 1 & \ddots & \vdots\\ \vdots & \vdots & \vdots & \cdots & a_{n-1}\\ 0 & \cdots & \cdots & \cdots & 1\end{bmatrix}\begin{bmatrix}CA^{n-1}\\ CA^{n-2}\\ \vdots\\ CA\\ C\end{bmatrix}\right]^{-1} \tag{9-146}$$

的线性变换，可以将系统转换成可观 II 型，其中 b_{o2} 由下列公式求得：

$$b_{o2}=T_{o2}^{-1}b=\begin{bmatrix}1 & a_{n-1} & \cdots & \cdots & a_1\\ 0 & 1 & a_{n-1} & \ddots & a_2\\ 0 & 0 & 1 & \ddots & \vdots\\ \vdots & \vdots & \vdots & \cdots & a_{n-1}\\ 0 & \cdots & \cdots & \cdots & 1\end{bmatrix}\begin{bmatrix}CA^{n-1}\\ CA^{n-2}\\ \vdots\\ CA\\ C\end{bmatrix}b \tag{9-147}$$

【例 9-13】 已知系统的状态空间描述为 $A=\begin{bmatrix}1 & 2 & 0\\ 3 & -1 & 1\\ 0 & 2 & 0\end{bmatrix}$，$B=\begin{bmatrix}2\\ 1\\ 1\end{bmatrix}$，$C=\begin{bmatrix}0 & 0 & 1\end{bmatrix}$，$D=0$，试求该系统的可观 I 型。

解：MATLAB 程序代码如下。

```
A=[1, 2, 0;3, -1, 1;0, 2, 0];
B=[2;1;1];
C=[0, 0, 1]
D=0
T=obsv(A, C)
[Ao1, Bo1, Co1, Do1]=ss2ss(A, B, C, D, T)
```

运行结果如下：

```
T=
     0     0     1
     0     2     0
     6    -2     2
Ao1=
     0     1     0
     0     0     1
    -2     9     0
Bo1=
     1
     2
    12
Co1=
```

```
       1    0    0  .
Do1 =
       0
```

由计算结果可知，该系统的可观Ⅰ型为$\begin{cases}\dot{Z}=\begin{bmatrix}0 & 1 & 0\\0 & 0 & 1\\-2 & 9 & 0\end{bmatrix}Z+\begin{bmatrix}1\\2\\12\end{bmatrix}U\\Y=\begin{bmatrix}1 & 0 & 0\end{bmatrix}Z\end{cases}$。

9.4　线性系统稳定性分析

9.4.1　稳定性分析基础

对于一个实际的控制系统，其工作的稳定性无疑是一个极其重要的问题，因为一个不稳定的系统在实际应用中是很难有效发挥作用的。

从直观上看，系统的稳定性就是一个处于稳态的系统，在某一干扰信号的作用下，其状态偏离了原有平衡位置，若系统是稳定的，那么在干扰取消后的有限时间内，系统会在自身作用下回到平衡状态；反之若系统不稳定，则系统永远不会回到原来的平衡位置。

系统的稳定分外部稳定和内部稳定两种。外部稳定又称做输出稳定，也就是系统在干扰取消后，在一定时间内其输出会恢复到原来的稳态输出。输出稳定有时描述为系统的 BIBO 稳定，即有限的系统输入只能产生有限的系统输出。系统内部稳定主要针对系统内部状态，反映的是系统内部状态受干扰信号的影响情况。当干扰信号取消后，若系统的内部状态会在一定时间内恢复到原来的平衡状态，则称系统状态稳定。

在经典控制论中，研究对象都是用高阶微分方程或传递函数描述的单输入单输出（SISO）系统，反映的仅是输入与输出的关系，不涉及系统的内部状态，因此经典控制论中只讨论系统的输出稳定问题。

系统的稳定性是系统本身的特性，与系统的外部输入（控制）无关。

如果系统不是线性定常系统，那么对于系统内部状态稳定问题，经典控制论中的方法就不好发挥作用了，这就需要用到下面介绍的李雅普诺夫（Lyapunov）稳定性理论。

设控制系统的齐次状态方程为：

$$\dot{X}=f(X,t)\qquad X(t)\big|_{t=t_0}=X_0 \tag{9-148}$$

其中，$X(t)$ 为系统的 n 维状态向量，f 是有关状态向量 X 以及时间 t 的 n 维矢量函数，f 不一定是线性定常的。如果对于所有 t，状态 X_e 总满足下式：

$$f(X_e,t)=0 \tag{9-149}$$

则称 X_e 为系统的平衡状态。对于一般控制系统，可能没有，也可能有一个或多个平衡状态。

如果系统是一个线性定常系统，即

$$\dot{X}=AX \tag{9-150}$$

那么当 $\boldsymbol{A}$ 为非奇异矩阵时，$\boldsymbol{X}_e=0$ 是系统的惟一平衡状态；当 $\boldsymbol{A}$ 为奇异矩阵时，$\boldsymbol{AX}=0$ 有无数解，也就是说系统有无数个平衡状态。

系统的状态稳定性是针对系统的平衡状态的，当系统有多个平衡状态时，需要对每个平衡状态分别进行讨论。对系统矩阵 $\boldsymbol{A}$ 非奇异的线性定常系统，$\boldsymbol{X}_e=0$ 是系统的惟一平衡状态，所以对线性定常（LTI）系统，一般可笼统地用 $\boldsymbol{X}_e$ 的稳定性代表系统的稳定性。

9.4.2　李雅普诺夫稳定性分析

1892 年，李雅普诺夫就如何判断系统的稳定性问题归纳出两种方法（称第一法和第二法），第一法的基本思路和分析方法与经典控制理论是一致的；第二法的特点是不求解系统方程，而是通过一个叫李雅普诺夫函数的标量函数来直控判断系统的稳定性。

9.4.2.1　李雅普诺夫稳定性定义

设一般控制系统的解为：

$$X(t)=\Phi(t;X_0,t_0) \tag{9-151}$$

它是与初始时间 t_0 及其初始状态 X_0 有关的，体现系统状态从 $(t_0,\boldsymbol{X}_0)$ 出发的一条状态轨迹。

设 $\boldsymbol{X}_e$ 为系统的一个平衡点，如果给定一个以 $\boldsymbol{X}_e$ 为球心，以 ε 为半径的 n 维球域 $S(\varepsilon)$，总能找到一个同样以 X_e 为球心，$\delta(\varepsilon,t_0)$ 为半径的 n 维球域 $S(\delta)$，使得从 $S(\delta)$ 球域出发的任意一条系统状态轨迹 $\Phi(t;X_0,t_0)$ 在 $t\geqslant t_0$ 的所有时间内，都不会跑出 $S(\varepsilon)$ 球域，则称系统的平衡状态 X_e 是李雅普诺夫稳定的（Lyapunov Stability）。

一般来说，δ 的大小不但与 ε 有关，而且与系统的初始时间 t_0 有关。当 δ 仅与 ε 有关时，称 $\boldsymbol{X}_e$ 是一致稳定的平衡状态。

进一步地，如果 X_e 不仅是李雅普诺夫稳定的平衡状态，而且当时间 t 无限增加时，从 $S(\delta)$ 球域出发的任一条状态轨迹 $\Phi(t;\boldsymbol{X}_0,t_0)$ 都最终收敛于球心平衡点 $\boldsymbol{X}_e$，那么称 X_e 是渐进稳定的（Asymptotic Stability）。

更进一步，如果从 $S(\infty)$ 即整个系统状态空间的任一点出发的任一条状态轨迹 $\Phi(t;\boldsymbol{X}_0,t_0)$，当 $t\to\infty$ 时都收敛于平衡点 X_e，那么称 X_e 是大范围渐进稳定的。显然，此时的 $\boldsymbol{X}_e$ 是系统的惟一平衡点。

反之，对于给定 $S(\varepsilon)$，不论 $\delta>0$ 取得多么小，若从 $S(\delta)$ 球域出发的状态轨迹 $\Phi(t;X_0,t_0)$ 至少有一条跑出 $S(\varepsilon)$ 球域，那么称平衡点 $\boldsymbol{X}_e$ 是不稳定的。

9.4.2.2　李雅普诺夫第一法（间接法）

李雅普诺夫第一法通过分析系统微分方程的显式解来分析系统的稳定性，对线性定常系统，它可以直控通过系统的特征根来分析。李雅普诺夫第一法的基本思路与经典控制论中的稳定性判别思路基本一致。

设线性定常系统的动态方程为：

$$\begin{cases}\dot{X}=\boldsymbol{AX}+\boldsymbol{b}U\\ \boldsymbol{Y}=\boldsymbol{CX}\end{cases} \tag{9-152}$$

在讨论系统状态稳定性（内部稳定）时，可以不考虑系统的输入结构和输入信号，而只考虑系统的齐次状态方程或矩阵 $\boldsymbol{A}$，显然，当$|\boldsymbol{A}|\neq 0$时，$\boldsymbol{X}_e=0$是系统的惟一平衡点。

对于$\boldsymbol{X}_e=0$的稳定性，有如下判据（$\boldsymbol{X}_e$大范围渐进输定的充分必要条件）：

当线性定常系统的系统矩阵 $\boldsymbol{A}$ 的所有特征根都有负的实部时，其惟一的状态平衡点$\boldsymbol{X}_e=0$是渐进稳定的，而且是大范围渐进稳定的。

对于式（9-152）所表示的系统，其输入/输出的传递函数为：

$$\boldsymbol{W}(s)=\boldsymbol{C}(s\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{b} \qquad (9\text{-}153)$$

当$\boldsymbol{W}(s)$的输点全部都有负实部时，该系统有界的输入将引起有界的输出（BIBO），也就是说系统是输出稳定的。

可以证明，当式（9-153）所表示的系统的传递函数$\boldsymbol{W}(s)$没有零极点对消时，系统的状态输定性和系统的输出稳定性是一致的，因为此时系统矩阵的特征根就是系统传递函数的极点。

9.4.2.3　李输普诺夫第二法（直接法）

李雅普诺夫第二法不必求解系统的状态方程，而是通过一个系统的能量函数来直接判断系统的输定性，所以又称直接法。直接法不但适用于线性定常系统，而且适用于非线性和时变的系统。

在实际系统中，往往不容易找出系统的能量函数，于是李雅普诺夫定义了一个正定的标量函数$\boldsymbol{V}(x)$，作为系统的虚构广义能量函数。根据$\dot{\boldsymbol{V}}(x)$的特号性质，可以判断系统的状态稳定性。

设$\boldsymbol{V}(x)$是定义在 n 维空间$\boldsymbol{R}_n$上的标量函数，且当$\boldsymbol{X}=0$时，$\boldsymbol{V}(x)=0$，那么对其余$\boldsymbol{X}\in\boldsymbol{R}_n$，有：

（1）若$\boldsymbol{V}(x)>0$，则称$\boldsymbol{V}(x)$是正定的；

（2）若$\boldsymbol{V}(x)\geqslant 0$，则称$\boldsymbol{V}(x)$是半正定的（非负定）；

（3）若$\boldsymbol{V}(x)<0$，称$\boldsymbol{V}(x)$是负定的；

（4）若$\boldsymbol{V}(x)\leqslant 0$，则称$\boldsymbol{V}(x)$是半负定的（非正定）；

（5）若$\boldsymbol{V}(x)$任意，则称$\boldsymbol{V}(x)$不定。

在建立于李雅普诺夫第二法基础上的稳定性分析中，有一类标量函数起着重要的作用，它就是二次型函数。

设

$$\boldsymbol{X}=(x_1,\ x_2,\ \cdots\ x_n)^{\mathrm{T}} \qquad (9\text{-}154)$$

$\boldsymbol{P}$ 为$n\times n$阶的实对称矩阵，则

$$\boldsymbol{V}(x)=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{X}$$

$$=\begin{bmatrix}x_1 & x_2 & \cdots & x_n\end{bmatrix}\begin{bmatrix}p_{11} & p_{12} & \cdots & p_{1n}\\ p_{21} & p_{22} & \cdots & p_{2n}\\ & & \cdots & \\ p_{n1} & p_{n2} & \cdots & p_{nn}\end{bmatrix}\begin{bmatrix}x_1\\ x_1\\ \\ x_n\end{bmatrix} \qquad (9\text{-}155)$$

称为二次型函数。

二次型标量函数的符号性质可以由赛尔维斯特（Sylvester）准则来判别。设实对称阵 $\boldsymbol{P}$ 的各阶主子行列式为：

$$\Delta_1 = p_{11},\ \Delta_2 = \begin{vmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{vmatrix} \cdots \cdots \Delta_n = |\boldsymbol{P}| \tag{9-156}$$

（1）若 $\Delta_i > 0\ (i = 1,2,\cdots,n)$，则 $V(x) > 0$；

（2）若 $\Delta_i \begin{cases} >0 & i\text{为偶数} \\ <0 & i\text{为奇数} \end{cases}$，或 $(-1)^i \Delta_i > 0$，则 $V(x) < 0$；

（3）若 $\Delta_i \begin{cases} \geqslant 0 & i = 1,2,\cdots,n-1 \\ =0 & i = n \end{cases}$，则 $V(x) \geqslant 0$；

（4）若 $\Delta_i \begin{cases} \geqslant 0 & i\text{为偶数} \\ \leqslant 0 & i\text{为偶数} \\ =0 & i = n \end{cases}$，则 $V(x) \leqslant 0$。

设系统状态方程为：

$$\dot{\boldsymbol{X}} = f(\boldsymbol{X}, t) \tag{9-157}$$

其中，$\boldsymbol{X}_e = 0$ 为系统的一个平衡状态。

如果存在一个正定的标量函数 $V(x)$，并且具有连续的一阶偏导数，那么根据 $\dot{V}(x) = \dfrac{\mathrm{d}V(x)}{\mathrm{d}t}$ 的符号性质，有：

（1）若 $\dot{V}(x) > 0$，则 $\boldsymbol{X}_e = 0$ 不稳定；

（2）若 $\dot{V}(x) \leqslant 0$，则 $\boldsymbol{X}_e = 0$ 李雅普诺夫稳定；

（3）若 $\dot{V}(x) < 0$ 或者 $\dot{V}(x) \leqslant 0$，当 $\boldsymbol{X} \neq 0$ 时 $\dot{V}(x)$ 不恒为零，则 $\boldsymbol{X}_e = 0$ 渐进稳定；

（4）若 $\boldsymbol{X}_e = 0$ 渐进稳定，且当 $\|\boldsymbol{X}\| \to \infty$ 时 $V(x) \to \infty$，则 $\boldsymbol{X}_e = 0$ 大范围渐进稳定。

应当指出，上述稳定性判据只是一个充分条件，并不是必要条件。如果给定的 $V(x)$ 满足上述 4 个条件之一，那么其结果成立。反之，如果给定的 $V(x)$ 不满足上述任何一个条件，那么只能说明所选的 $V(x)$ 对式（9-157）所表示的系统失效，必须重新构造 $V(x)$。

9.4.2.4　线性定常系统的李雅普诺夫稳定分析及系统参数优化

研究下列线性定常系统：

$$\dot{\boldsymbol{X}} = \boldsymbol{A}\boldsymbol{X} \tag{9-158}$$

若 $\boldsymbol{A}$ 为非奇异矩阵，那么 $\boldsymbol{X}_e = 0$ 是系统的惟一平衡状态，其稳定性可以通过李雅普诺夫第二法来分析。

取

$$V(\boldsymbol{X}) = \boldsymbol{X}^{\mathrm{T}} \boldsymbol{P} \boldsymbol{X} \tag{9-159}$$

其中，$\boldsymbol{P}$ 为正定实对称矩阵，所以 $V(x)$ 对 $\boldsymbol{X}$ 有连续偏导数，并且 $V(x) > 0$。

$$\dot{V}(x)=\dot{X}^{\mathrm{T}}\boldsymbol{P}X+X^{\mathrm{T}}\boldsymbol{P}\dot{X}=(AX)^{\mathrm{T}}\boldsymbol{P}X+X^{\mathrm{T}}\boldsymbol{P}(AX)$$
$$=X^{\mathrm{T}}A^{\mathrm{T}}\boldsymbol{P}X+X^{\mathrm{T}}\boldsymbol{P}AX=X^{\mathrm{T}}(A^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}A)X$$

令：

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}=-\boldsymbol{Q} \tag{9-160}$$

式（9-160）称做李雅普诺夫方程。

可得：

$$\dot{V}(X)=-X^{T}QX \tag{9-161}$$

其中，$\boldsymbol{Q}=-(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A})$ 为对称矩阵。若 $\boldsymbol{Q}>0$，则 $\dot{V}(x)<0$，因此 $X_e=0$ 为渐进稳定，而且是大范围渐进稳定。

在实际应用中，先给定一正定矩阵 $\boldsymbol{Q}$，然后通过李雅普诺夫方程式（9-160）求出实对称矩阵 $\boldsymbol{P}$，最后通过赛尔维斯特准则判别 $\boldsymbol{P}$ 的正定性。若 $\boldsymbol{P}>0$，则系统稳定。

在应用李雅普诺夫方程时，应注意以下几点：

（1）由李雅普诺夫方程求得的 $\boldsymbol{P}$ 为正定是 $X_e=0$ 渐进稳定的充分必要条件。

（2）$\boldsymbol{Q}$ 的选取是任意的，只要满足对称且正定（在一定条件下可以是半正定），$\boldsymbol{Q}$ 的选取不会影响系统稳定性判别的结果。

（3）如果 $a_t(t)$ 沿任意一条轨迹不恒等于零，那么 $\boldsymbol{Q}$ 可以取半正定阵，即 $\boldsymbol{Q}\geqslant 0$。

（4）当 $\boldsymbol{Q}$ 取为单位阵 $\boldsymbol{I}$ 时，李雅普诺夫方程变为：

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}=-\boldsymbol{I} \tag{9-162}$$

这是一个比较简单的李雅普诺夫方程。

MATLAB 提供了求解李雅普诺夫方程的函数 lyap( ), lyap2( ), dlyap( )，函数的调用格式如下：

X＝lyap(A, C)，其中输入参数 A 是已知系统的状态矩阵，C 是给定的正定对称矩阵，输出量 X 是李雅普诺夫方程的解，即正定实对称矩阵 P。

X＝dlyap(A, C)，其中输入参数 A 是已知离散系统的状态矩阵，C 是给定的正定对称矩阵，输出量 X 是离散李雅普诺夫方程的解，即正定实对称矩阵 P。

X＝lyap2(A, C)，其中输入参数 A 是已知系统的状态矩阵，C 是给定的正定对雅矩阵，输出量 X 是李雅普诺夫方雅的解，即正定实对称矩阵 P。lyap2( )采用特征值分解法求解李雅普诺夫方程，其运算速度比 lyap( )快很多。

【例 9-14】 已知系统的状态矩阵 $\boldsymbol{A}=\begin{bmatrix}0 & 1\\-1 & -1\end{bmatrix}$，给定的正定对称矩阵 $\boldsymbol{C}=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$，试求李雅普诺夫方程的解 P。

解：MATLAB 程序代码如下。

```
A=[0, 1; -1, -1]
C=[1, 0; 0, 1]
X=lyap(A', C)
```

运行结果如下：

```
X=
      1.5000    0.5000
      0.5000    1.0000
```

由运算结果可知，李雅普诺夫方程的解 $\boldsymbol{P}=\begin{bmatrix}1.5 & 0.5\\0.5 & 1.0\end{bmatrix}$。

第 10 章　线性系统状态空间设计

10.1　引　　言

控制系统分析（System Analysis）与综合设计（System Synthesis）是系统研究中的两大课题。系统分析是在建立控制系统数学模型的基础上，分析系统的各种性能。系统综合或系统设计的任务则是通过设计系统控制器，改善原有系统的性能，从而更好地达成系统所要求的各种性能指标。

状态空间法适用于多输入多输出（MIMO）系统，它根据给定的性能指标设计系统，而不必根据特定的输入函数（脉冲函数、阶跃函数）进行设计，同时还可以包含初始条件，且状态空间法还可用于非线性控制系统与时变系统的设计，如最优控制系统、自适应控制系统。

通过本章，读者对线性系统状态空间设计的理论和方法有全面的认识，并熟练使用 MATLAB 进行状态空间设计。

10.2　状态反馈与极点配置

与经典控制理论一样，现代控制系统中仍然主要采用反馈控制结构，但不同的是，经典控制理论中主要采用输出反馈，而现代控制系统中主要采用内部状态反馈。状态反馈可以为系统控制提供更多的信息反馈，从而实现更优的控制。

闭环系统极点的分布情况决定于系统的稳定性和动态品质，因此，可以根据对系统动态品质的要求，规定闭环系统的极点所应具备的分布情况，把极点的布置作为系统的动态品质指标。这种把极点布置在希望的位置的过程称为极点配置。在空间状态法中，一般采用反馈系统状态变量或输出变量的方法，来实现系统的极点配置。

10.2.1　状态反馈

状态反馈是将系统的内部状态变量乘以一定的反馈系数（矢量），然后反馈到系统输入端与系统的参考输入综合，综合而成的信号作为系统的输入对系统实施控制。

控制系统结构如图 10.1 所示，实线部分表示原来系统 $\sum(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D})$ 的结构图，其动态方程为：

$$\begin{cases}\dot{\boldsymbol{X}}=\boldsymbol{AX}+\boldsymbol{BU}\\ \boldsymbol{Y}=\boldsymbol{CX}+\boldsymbol{DU}\end{cases}\tag{10-1}$$

当加上图中虚线所示的状态反馈环节后，其中的线性状态反馈控制律为：

$$u=\boldsymbol{R}+\boldsymbol{K}x\tag{10-2}$$

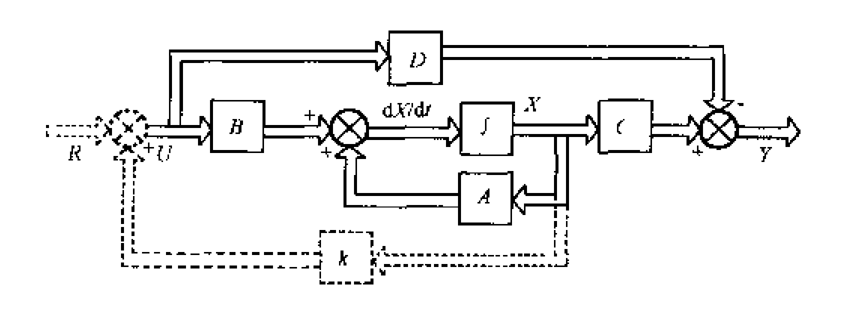


图 10.1　状态反馈控制结构图

式中，$\boldsymbol{R}$ 是参考输入，$\boldsymbol{K}$ 称为状态反馈增益矩阵，为 $p\times n$ 矩阵。

系统动态方程变为：

$$\begin{cases}\dot{\boldsymbol{X}}=\boldsymbol{AX}+\boldsymbol{B}(\boldsymbol{KX}+\boldsymbol{R})=(\boldsymbol{A}+\boldsymbol{BK})\boldsymbol{X}+\boldsymbol{BR}=\boldsymbol{A}_K\boldsymbol{X}+\boldsymbol{BR}\\ \boldsymbol{Y}=\boldsymbol{CX}+\boldsymbol{D}(\boldsymbol{KX}+\boldsymbol{R})=(\boldsymbol{C}+\boldsymbol{DK})\boldsymbol{X}+\boldsymbol{DR}=\boldsymbol{C}_K\boldsymbol{X}+\boldsymbol{DR}\end{cases}\tag{10-3}$$

其中，$\boldsymbol{A}_K=\boldsymbol{A}+\boldsymbol{BK}$，$\boldsymbol{C}_K=\boldsymbol{C}+\boldsymbol{DK}$，当 $\boldsymbol{D}=0$ 时，状态反馈系统闭环传递函数 $W_K(s)$ 为：

$$\boldsymbol{W}_K(s)=\boldsymbol{C}[s\boldsymbol{I}-(\boldsymbol{A}+\boldsymbol{BK})]^{-1}\boldsymbol{B}\tag{10-4}$$

式中，$\boldsymbol{A}+\boldsymbol{BK}$ 为闭环系统的系统矩阵。

从式（10-1）和式（10-3）可以看出，状态反馈前后的系统矩阵分别为 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{BK}$，特征方程分别为 $\det[\lambda\boldsymbol{I}-\boldsymbol{A}]$ 和 $\det[\lambda\boldsymbol{I}-(\boldsymbol{A}+\boldsymbol{BK})]$，可看出状态反馈后的系统特征根（即系统的极点）不仅与系统本身的结构参数有关，而且与状态反馈 $\boldsymbol{K}$ 有关。应该指出完全能控的系统经过状态反馈后，仍是完全能控的，但状态反馈可能改变系统的能观性，即原来可观的系统在某些状态反馈下，闭环可以是不可观的。同样，原来不可观的系统在某些状态反馈下，闭环可以是可观的。状态反馈是否改变系统的可观测性，要做具体分析。

状态反馈后的控制系统 $\sum_K(\boldsymbol{A}_K,\boldsymbol{B},\boldsymbol{C}_K,\boldsymbol{D})$ 其系统维数不变，但系统矩阵 $\boldsymbol{A}_K$ 及系统输出矩阵 $\boldsymbol{C}_K$ 随反馈环节 $\boldsymbol{K}$ 而改变。通过调整 $\boldsymbol{K}$ 可以改善系统的稳定性、快速性、稳定误差，以及系统可观性与可控性，这也是后面利用状态反馈对极点进行配置的依据。

10.2.2　输出反馈

把系统的输出变量按照一定的比例关系反馈到系统的输入端或 $\dot{\boldsymbol{X}}$ 端称为输出反馈，如图 10.2 所示。由于状态变量不一定具有物理意义，所以状态反馈往往不易实现；而输出变量则具有明显的物理意义，因此输出反馈比较容易实现。

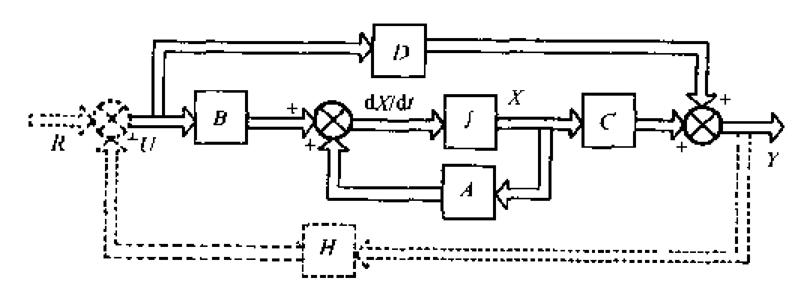


图 10.2　输出反馈控制结构图

输出反馈是采用输出矢量 $\boldsymbol{Y}$ 构成反馈信号综合到系统的控制输入端，以改善系统的各项性能。

实线部分表示原来系统 $\sum(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D})$，虚线部分表示加上输出反馈环节 $\boldsymbol{H}$ 后形成的输出反馈。输出反馈的控制信号为：

$$\boldsymbol{U}=\boldsymbol{HY}+\boldsymbol{R}=\boldsymbol{H}(\boldsymbol{CX}+\boldsymbol{DU})+\boldsymbol{R}=\boldsymbol{HCX}+\boldsymbol{HDU}+\boldsymbol{R} \tag{10-5}$$

化简得：

$$\boldsymbol{U}=(\boldsymbol{I}-\boldsymbol{HD})^{-1}(\boldsymbol{HCX}+\boldsymbol{R}) \tag{10-6}$$

将式（10-6）代入式（10-1）所代表的原系统动态方程，得：

$$\begin{cases}\dot{\boldsymbol{X}}=\boldsymbol{AX}+\boldsymbol{B}(\boldsymbol{I}-\boldsymbol{HD})^{-1}(\boldsymbol{HCX}+\boldsymbol{R})=\boldsymbol{A}_H\boldsymbol{X}+\boldsymbol{B}_H\boldsymbol{R}\\ \boldsymbol{Y}=\boldsymbol{CX}+\boldsymbol{D}(\boldsymbol{I}-\boldsymbol{HD})^{-1}(\boldsymbol{HCX}+\boldsymbol{R})=\boldsymbol{C}_H\boldsymbol{X}+\boldsymbol{D}_H\boldsymbol{R}\end{cases} \tag{10-7}$$

其中，$\boldsymbol{A}_H=\boldsymbol{A}+\boldsymbol{B}(\boldsymbol{I}-\boldsymbol{HD})^{-1}\boldsymbol{HC}$，$\boldsymbol{B}_H=\boldsymbol{B}(\boldsymbol{I}-\boldsymbol{HD})^{-1}$，$\boldsymbol{C}_H=\boldsymbol{C}+\boldsymbol{D}(\boldsymbol{I}-\boldsymbol{HD})^{-1}\boldsymbol{HC}$，$\boldsymbol{D}_H=\boldsymbol{D}(\boldsymbol{I}-\boldsymbol{HD})^{-1}$。

当 $\boldsymbol{D}=0$ 时，

$$\begin{cases}\boldsymbol{A}_H=\boldsymbol{A}+\boldsymbol{BHC}\\ \boldsymbol{B}_H=\boldsymbol{B}\\ \boldsymbol{C}_H=\boldsymbol{C}\\ \boldsymbol{D}_H=0\end{cases} \tag{10-8}$$

输出反馈系统闭环传递函数 $\boldsymbol{W}_H(s)$ 为：

$$\boldsymbol{W}_H(s)=\boldsymbol{C}[s\boldsymbol{I}-(\boldsymbol{A}+\boldsymbol{BHC})]^{-1}\boldsymbol{B} \tag{10-9}$$

式中，$\boldsymbol{A}+\boldsymbol{BHC}$ 为闭环系统的系统矩阵。

从式（10-1）和式（10-7）可以看出，输出反馈前后的系统矩阵分别为 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{BHC}$，特征方程分别为 $\det[\lambda\boldsymbol{I}-\boldsymbol{A}]$ 和 $\det[\lambda\boldsymbol{I}-(\boldsymbol{A}+\boldsymbol{BHC})]$，可看出状态反馈后的系统特征根（即系统的极点）不仅与系统本身的结构参数有关，而且与输出反馈矩阵 $\boldsymbol{H}$ 有关。也就是说，输出反馈 $\boldsymbol{H}$ 改变了原系统的系统矩阵 $\boldsymbol{A}$，从而改变了系统稳定性（改变了系统的特征根）、系统可控性（改变了可控判别阵 $\boldsymbol{M}$）和系统可观性（改变了可观判别矩阵 $\boldsymbol{N}$）。

比较式（10-4）和式（10-9）可知，输出反馈系统中的 $\boldsymbol{HC}$ 相当于状态反馈中的 K。但由于 $\boldsymbol{H}$ 是 $r\times m$ 阶矩阵，而 $\boldsymbol{K}$ 是 $r\times n$ 阶矩阵（其中 r 是系统输入维数，m 是输出维数，n 是状态维数），且通常情况下 $n>m$，因此状态反馈一般能提供更多的系统反馈信息，从而带来更优的控制效果。

从上面的分析中可以归纳出状态反馈与输出反馈的如下基本特点：

（1）两种形式反馈的重要特点是反馈的引入并不增加新的状态变量，即闭环系统和开环系统具有相同的阶数。

（2）两种反馈闭环系统均能保持反馈引入前的能控性，而对于反馈闭环系统的能观性则不然。对状态反馈，闭环后不一定能保持系统的能观性；而对输出反馈，闭环后必定能保持系统的能观性。

（3）在工程实现方面，两种反馈形式都会遇到一定的困难。

（4）输出反馈的一个突出优点是工程上构成方便，但事实证明，状态反馈比输出反

馈具有更好的特性。对具体系统而言，要从实际出发进行具体分析与选择。

10.2.3　极点配置

动力学的各种特性或各种品质指标，在很大程度上是由系统的极点决定的，因此系统设计的一个重要目标是在 S 平面上设计一组系统所希望的极点。

所谓极点配置问题，就是通过反馈矩阵的选择，使闭环系统的极点，即闭环特征方程的特征值恰好处于所希望的一组极点位置上。由于希望的极点具有一定的任意性，因此极点的配置也具有一定的任意性。

状态反馈和输出反馈（主要指输出反馈至 $\dot{X}$ 的情况）都能够对系统进行极点配置，且一般认为用简单的比例反馈就能使问题得到解决。

极点配置是通过选择一个状态反馈矩阵，使闭环系统的极点处于期望的位置上。在状态空间中，极点任意配置的充分必要条件是系统必须是完全状态可控的。

极点配置方法如下所述：如果系统是完全状态可控的，那么可选择期望设置的极点，然后以这些极点作为闭环极点来设计系统，利用状态观测器反馈全部或部分状态变量，使所有的闭环极点均落在各期望位置上，以满足系统的性能要求。这种设置期望闭环极点的方法就称为极点配置方法。

在极点配置方法中，为使全部的闭环极点位于期望的位置上，需要反馈全部的状态变量。但在实际系统中，不可能测量到全部的状态变量，为了实现状态反馈，利用状态观测器对未知的状态变量进行估计是十分必要的。

设给定的线性定常系统为：

$$\dot{\boldsymbol{X}} = \boldsymbol{AX} + \boldsymbol{BU} \tag{10-10}$$

式中，$\boldsymbol{X}$ 为 n 维状态向量；$\boldsymbol{U}$ 为 p 维状态向量；$\boldsymbol{A}$ 和 $\boldsymbol{B}$ 为相应维数的常数阵。若给定 n 个反馈性能的期望闭环极点为：

$$\{p_1, p_2, \cdots, p_n\} \tag{10-11}$$

则极点配置的设计问题就是确定一个 $p\times n$ 状态反馈增益矩阵 $\boldsymbol{K}$，使状态反馈闭环系统

$$\dot{\boldsymbol{X}} = (\boldsymbol{A} - \boldsymbol{BK})\boldsymbol{X} + \boldsymbol{B}v \tag{10-12}$$

的极点为 $\{p_1, p_2, \cdots, p_n\}$，即

$$\lambda_i(\boldsymbol{A} - \boldsymbol{BK}) = p_i (i = 1, 2, \cdots, n) \tag{10-13}$$

其中，$\lambda_i(\cdot)$ 表示 $(\cdot)$ 的特征值。

10.2.3.1　单输入单输出系统的极点配置

对于单输入单输出的 n 阶系统，其反馈增益矩阵 K 是一行向量，仅包含 n 个元素，可由 n 个极点惟一确定。反馈增益矩阵 $\boldsymbol{K}$ 由期望的闭环极点确定，而期望的闭环极点根据闭环系统的设计要求决定，如对响应速度、阻尼比、带宽等的要求。

设已知系统为 $\Sigma(\boldsymbol{A}, \boldsymbol{b})$，期望的闭环极点为 $\{p_1, p_2, \cdots, p_n\}$，要确定 $1\times n$ 的反馈增益矩阵 $\boldsymbol{K}$，使得 K 满足式（10-13）。

单输入单输出系统极点配置方法设计步骤如下：

（1）确定受控系统$\Sigma(\boldsymbol{A},\boldsymbol{b})$完全可控，如果系统不是完全可控，则不能进行极点配置，并确定系统开环特征多项式 det(*sI*–*A*)。

$$\det(s\boldsymbol{I}-\boldsymbol{A})=s^n+a_{n-1}s^{n-1}+\cdots+a_1s+a_0 \tag{10-14}$$

（2）由希望的闭环极点$\{p_1,p_2,\cdots,p_n\}$计算闭环期望的特征多项式。

$$\det[\lambda\boldsymbol{I}-(\boldsymbol{A}+\boldsymbol{BK})]=(s-p_1)(s-p_2)\cdots(s-p_n)=s^n+a'_{n-1}s^{n-1}+\cdots+a'_1s+a'_0 \tag{10-15}$$

（3）计算

$$\bar{\boldsymbol{K}}=\boldsymbol{KP}=[a'_0-a_0 \quad a'_1-a_1 \quad \cdots \quad a'_{n-1}-a_{n-1}] \tag{10-16}$$

（4）计算变换矩阵$\boldsymbol{P}$及其逆$\boldsymbol{P}^{-1}$，

$$\boldsymbol{P}=\begin{bmatrix}\boldsymbol{A}^{n-1}\boldsymbol{b} & \cdots & \boldsymbol{Ab} & \boldsymbol{b}\end{bmatrix}\begin{bmatrix}1 & 0 & \cdots & 0\\ a_{n-1} & \ddots & & \vdots\\ \vdots & & \ddots & 0\\ a_1 & \cdots & a_{n-1} & 1\end{bmatrix} \tag{10-17}$$

（5）将所求出的状态反馈增益阵转换成实际实施的$\boldsymbol{K}$，$\boldsymbol{K}=\bar{\boldsymbol{K}}\boldsymbol{P}^{-1}$。

MATLAB 提供了进行极点配置的函数 acker( )和 place( )，它们的调用格式如下：

K=acker(A, B, P)，其中 A，B 为系统系数矩阵，P 为配置的极点，K 为反馈增益矩阵。

K=place(A, B, P)，其中 A，B 为系统系数矩阵，P 为配置的极点，K 为反馈增益矩阵。

[K, prec, message]=place(A, B, P)，其中 A，B 为系统系数矩阵，P 为配置的极点，K 为反馈增益矩阵，Prec 为特征值，message 为配置时的报错信息。

命令 acker 基于爱克曼公式（Ackerman），只适用于单输入系统，希望的极点可以包括多重极点（位于同一位置的多个极点）。

Ackerman 公式如下所述。

若单输入系统是可控的，那么反馈矩阵$\boldsymbol{K}$可由式（10-18）求得。

采用的状态反馈规律为$u=r+\boldsymbol{K}x$，则反馈矩阵$\boldsymbol{K}$为：

$$\boldsymbol{K}=[0 \quad \cdots \quad 0 \quad 1]S^{-1}\Lambda \tag{10-18}$$

式中，S 式系统的可控性矩阵为：

$$\Lambda=\boldsymbol{A}^n+a_0^*\boldsymbol{A}^{n-1}+a_1^*\boldsymbol{A}^{n-2}+\cdots+a_{n-2}^*\boldsymbol{A}+a_{n-1}^*\boldsymbol{I} \tag{10-19}$$

其中，$\boldsymbol{A}$ 是系统矩阵，a_i^* 是所需闭环特征方程的系数。

$$s^n+a_0^*s^{n-1}+a_1^*s^{n-2}+\cdots+a_{n-2}^*s+a_{n-1}^*=0 \tag{10-20}$$

【例 10-1】 已知系统的方程为：$\boldsymbol{X}=\boldsymbol{AX}+\boldsymbol{BU}$，其中

$$\boldsymbol{A}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -1 & -5 & -6\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}$$

采用状态反馈$\boldsymbol{U}=-\boldsymbol{KX}$，希望的闭环极点为$p_{1,2}=-2\pm j4$，$p_3=-10$，试用 MATLAB 确定状态反馈增益矩阵，并计算当系统初始条件为$\boldsymbol{X}_0=[1,\ 0,\ 0]$时的响应。

解：MATLAB 程序代码如下。

```
%状态矩阵 A
A=[0, 1, 0; 0, 0, 1; -1, -5, -6]
%输入矩阵 B
B=[0; 0; 1]
%希望配置的极点
P=[-2+j*4, -2-4*j, -10]
%进行极点配置
K=acker(A, B, P)
sys_new=ss(A-B*K, eye(3), eye(3), eye(3))
t=0:0.1:4
%初始条件
X=initial(sys_new, [1; 0; 0], t)
x1=[1, 0, 0]*X'
x2=[0, 1, 0]*X'
x3=[0, 0, 1]*X'
subplot(3, 1, 1)
plot(t, x1)
grid
title('零输入响应')
ylabel('状态变量 x1')
subplot(3, 1, 2)
plot(t, x2)
grid
ylabel('状态变量 x2')
subplot(3, 1, 3)
plot(t, x3)
grid
xlabel('时间/秒')
ylabel('状态变量 x3')
```

运行结果如下：

```
%状态反馈增益矩阵
K=
   199    55    8
```

由输出结果可知，所求状态反馈增益矩阵 $\boldsymbol{K}=[199,55,8]$。

响应曲线如图 10.3 所示。

【例 10-2】　已知数字控制系统的状态方程为：

$$x(k+1)=\begin{bmatrix}0 & 1\\ -0.16 & -1\end{bmatrix}x(k)+\begin{bmatrix}0\\ 1\end{bmatrix}u(k)$$

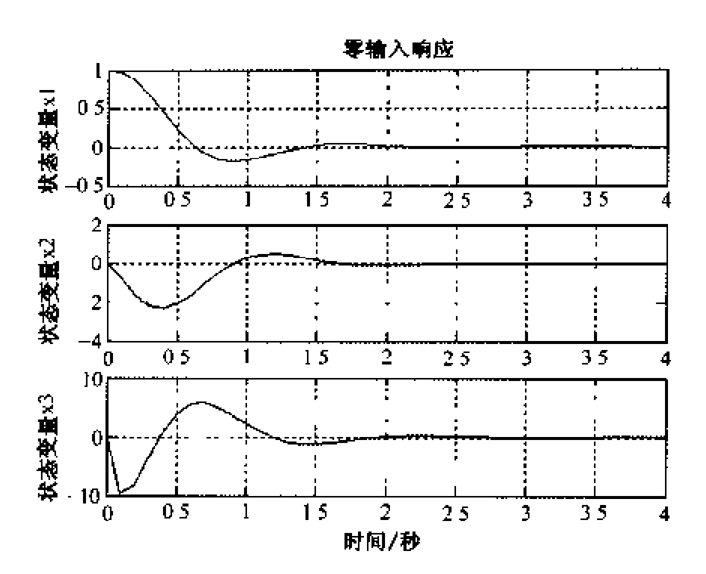


图 10.3　零输入条件下系统状态的响应曲线

设系统期望的闭环极点为 $z=0.5\pm j0.5$，现用全状态反馈控制系统，求反馈增益矩阵 $\boldsymbol{K}$。

解：MATLAB 程序代码如下。

```
A=[0, 1; -0.16, -1]
B=[0; 1]
%希望的极点
P=[0.5+j*0.5, 0.5-0.5*j]
%进行极点配置
K=acker(A, B, P)
```

运行结果如下：

```
%状态反馈增益矩阵
K=
    0.3400    -2.0000
```

由输出结果可知，所求状态反馈增益矩阵 $\boldsymbol{K}=[0.34,-2]$。

【例 10-3】　已知如图 10.4 所示的受控系统，其中 $G_1(s)=\dfrac{1}{s}$，$G_2(s)=\dfrac{1}{s+6}$，$G_3(s)=\dfrac{1}{s+12}$，状态变量 x_1,x_2,x_3 如图 10.4 所示，试对系统进行极点配置，以达到系统期望的指标：输出超调量 $\sigma_p\leqslant 5\%$；超调时间 $t_p\leqslant 0.5\,\text{s}$；系统频宽 $\omega_b\leqslant 10$；跟踪误差 $e_p=0$（对于阶跃输入）、$e_v\leqslant 0.2$（对于速度输入），并求极点配置后系统的阶跃响应。

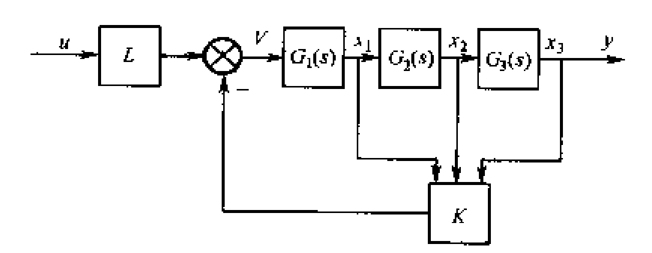


图 10.4　例 10-3 的系统结构图

解：

（1）确定受控系统的状态空间模型。

由图可知：$x_1(s)=\boldsymbol{V}(s)\boldsymbol{G}_1(s)$，$x_2(s)=x_1(s)\boldsymbol{G}_2(s)$，$x_3(s)=x_2(s)\boldsymbol{G}_3(s)$，$y=x_3$，把题中的条件代入，得系统的状态方程为：

$$\begin{cases}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix}=\begin{bmatrix}0 & 0 & 0\\ 1 & -6 & 0\\ 0 & 1 & -12\end{bmatrix}\begin{bmatrix}x_1\\ x_3\\ x_3\end{bmatrix}+\begin{bmatrix}1\\ 0\\ 0\end{bmatrix}v\\ y=\begin{bmatrix}0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}\end{cases}$$

（2）确定希望的极点。

由于系统是三阶系统，系统有 3 个极点，可选定其中一对为主导极点，另一个为远极点，系统的性能主要由主导极点决定，远极点对系统的影响不大。

根据二阶系统的关系，先求出主导极点：

$$\sigma_p=\mathrm{e}^{-\frac{\pi\xi}{\sqrt{1-\xi^2}}}$$

$$t_p=\frac{\pi}{\omega_n\sqrt{1-\xi^2}}$$

$$\omega_b=\omega_n\left(\sqrt{1-2\xi^2+\sqrt{2-4\xi^2+4\xi^4}}\right)$$

式中，ξ 和 ω_n 为此二阶系统的阻尼比和自然频率。

由 $\sigma_p=\mathrm{e}^{-\frac{\pi\xi}{\sqrt{1-\xi^2}}}\leqslant 5\%$，可得 $-\frac{\pi\xi}{\sqrt{1-\xi^2}}\geqslant 3.14$，从而有 $\xi\geqslant\frac{\sqrt{2}}{2}$，选 $\xi=\frac{\sqrt{2}}{2}$。

由 $t_p\leqslant 0.5\,\mathrm{s}$ 得：

$$\frac{\pi}{\omega_n\sqrt{1-\xi^2}}\leqslant 0.5$$

$$\omega_n\geqslant\frac{\pi}{0.5\times\frac{\sqrt{2}}{2}}\approx 9$$

由 $\omega_b\leqslant 10$ 和 $\xi=\frac{\sqrt{2}}{2}$ 得 $\omega_n=10$，这样，主导极点为：

$$p_{1,2}=-\xi\omega_n\pm\mathrm{j}\omega_n\sqrt{1-\xi^2}$$

远极点选择使得它和原点的距离大于 $5|p_1|$，现取 $p_3=10|p_1|$，因此确定的希望极点为：

$$\begin{cases}p_1=-5\sqrt{2}+\mathrm{j}5\sqrt{2}\\ p_2=-5\sqrt{2}-\mathrm{j}5\sqrt{2}\\ p_3=-100\end{cases}$$

（3）确定状态反馈矩阵 $\boldsymbol{K}$。

MATLAB 程序如下：

```
A=[0, 0, 0; 1, -6, 0; 0, 1, -12]
B=[1; 0; 0]
%希望的极点
P=[-sqrt(2)*5+j*5*sqrt(2), -sqrt(2)*5-j*5*sqrt(2), -100]
%进行极点配置
K=place(A, B, P)
```

运行结果如下：

```
%状态反馈增益矩阵
K=
    96.14, -288.3, 6538.
```

由输出结果可知，所求状态反馈增益矩阵为：

$$\boldsymbol{K}=[96.14,-288.3,6538]$$

（4）确定输入放大系数。

对应的闭环传递函数为 $W_K(s)=\dfrac{L}{s^3+114.1s^2+1510s+10000}$。

由于系统要求的跟踪阶跃信号误差为 0，则

$$e_p=0=\lim_{t\to\infty}(1-y(t))=\lim_{s\to 0}s\left(\frac{1}{s}-\frac{W_K(s)}{s}\right)=\frac{10000-L}{10000}$$

得

$$L=10000$$

（5）求极点配置后系统的阶跃响应。

MATLAB 的程序如下：

```
A=[0, 0, 0; 1, -6, 0; 0, 1, -12]
B=[1; 0; 0]
C=[0, 0, 1]
D=0
P=[-sqrt(2)*5+j*5*sqrt(2), -sqrt(2)*5-j*5*sqrt(2), -100]
K=place(A, B, P)
%输入放大系数
L=10000
sys_new=ss(A-B*K, B, C, D)
t=0:0.01:5
y_new=L*step(sys_new, t)
plot(t, y_new)
grid
xlabel('时间/秒')
```

```
ylabel('y(t)')
```

程序运行后，输出结果如图 10.5 所示。

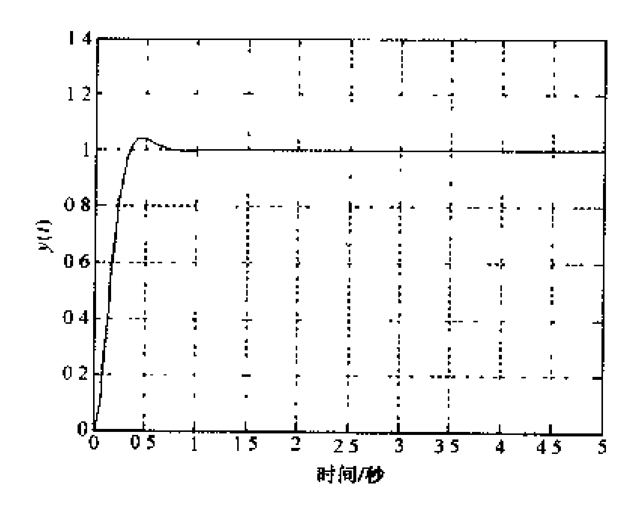


图 10.5　极点配置后系统的阶跃响应曲线

10.2.3.2　多输入多输出系统的极点配置

对于多输入多输出系统的极点配置，有多种算法可以确定状态反馈增益矩阵，但与单输入单输出系统相比，则要复杂得多。在多变量系统中，状态反馈增益矩阵 $\boldsymbol{K}$ 不是惟一的，如果要确定惟一的极点，则必须附加其他条件，但从工程应用角度来说，希望 $\boldsymbol{K}$ 的各个元素尽可能小。

多输入多输出系统 $\Sigma(\boldsymbol{A}, \boldsymbol{B})$ 极点配置方法设计步骤如下所述：

（1）将能控矩阵对 $\{\boldsymbol{A},\boldsymbol{B}\}$ 转化成某种规范型（如龙博格规范型）；

（2）将给定的期望闭环极点 $\{p_1, p_2, \cdots, p_n\}$ 按规范型 $\bar{\boldsymbol{A}}$ 计算它们的特征多项式；

（3）求取规范型的状态反馈增益阵 $\boldsymbol{K}$。

10.2.3.3　极点配置的注意问题

使用极点配置方法时，要注意以下问题：

（1）系统完全状态可控是求解的充分必要条件；

（2）应把闭环系统的期望特性转化为极点位置；

（3）理论上，选择反馈增益可使系统有任意快的时间响应。加大反馈增益可提高系统的频带、加快系统的响应。但过大的反馈增益，在有一定误差信号时，将会导致控制信号无限增大，这在工程上是无法实现的，因此必须考虑到反馈增益物理实现的可能性。

（4）当系统的阶次较高时，可用 Ackerman 公式，通过计算机求解。

10.3　状态观测器

在前面介绍控制系统设计中的极点配置方法时，曾假设所有的状态变量均可有效地用于反馈，而在实际工作中，并不是所有的状态变量都可用于反馈的，这时就需要估计

不可观测的状态变量。不可观测状态变量的估计通常称为观测，估计状态变量的装置或算法称为状态观测器，或简称观测器。

10.3.1　状态观测器的基本概念

10.3.1.1　状态观测器定义

对于完全能控的线性定常系统，通过状态反馈配置期望的极点，使闭环系统具有期望要求的动态特性。但在实际系统中，并不是所有的状态变量都能测量到的。但为了实现状态反馈控制律，必须对状态变量进行测量，因此要设法利用已知的信息（输入量与输出量），通过一个模型（或系统、软件）来对状态变量进行估计。为了解决这一问题，提出了状态重构问题。

龙博格（Luenberger）提出了状态观测器理论，它是现代控制理论中具有工程实用价值的基本内容之一，这个理论解决了在确定性控制条件下受控系统状态的重构问器，从而使状态反馈成为一种现实的控制规律。

重构状态就是在系统的实际状态不可得的情况下构造系统，利用原系统可直接测量的输入变量 u 和输出变量 y 重新构造一个状态 $\hat{x}(t)$，使之在一定的指标下和系统的真实状态 $x(t)$ 等价，即 $\lim\limits_{t\to\infty}\hat{x}(t)=\lim\limits_{t\to\infty}x(t)$ 。

如果系统可观测，从可测量 u 和 y 中把 $x(t)$ 间接重构出来是可能的，这种必要性与可能性正是观测器理论的出发点。状态观测器又称为状态渐近估计器。

设线性定常系统 $\Sigma_0=(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C})$ 的状态 $\boldsymbol{X}$ 是不能直接测量的，如果动态系统 $\hat{\Sigma}$ 以 Σ_0 的输入 u 和输出 y 作为它的输入量，$\hat{\Sigma}$ 的输出 $\hat{x}(t)$ 满足如下等价性指标：

$$\lim_{t\to\infty}\left[x(t)-\hat{x}(t)\right]=0 \tag{10-21}$$

则称动态系统 $\hat{\Sigma}$ 为 Σ_0 的状态观测器。

10.3.1.2　状态观测器的构成

一般系统的输入量 u 和输出量 y 均为已知，因此希望利用 $y=cx$ 与 $\hat{y}=c\hat{x}$ 的偏差信号来修正 $\hat{x}$ 的值，这样就形成了如图 10.6 所示的闭环估计方案。

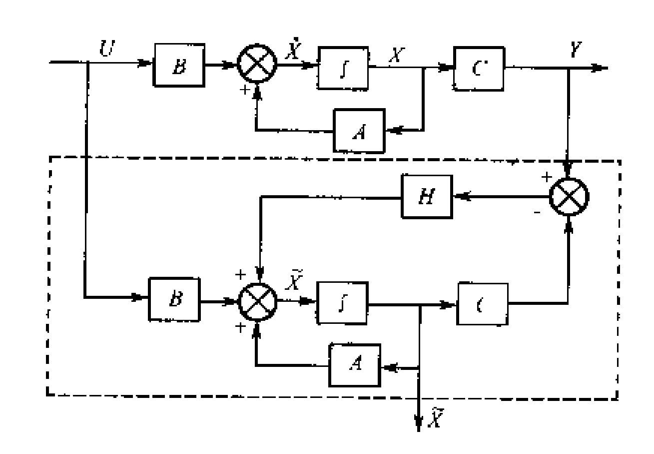


图 10.6　状态观测器结构图

在图 10.6 中，虚线框出的部分称为状态观测器或状态估计器，它是一个动态系统，以原系统的输入量和输出量作为它的输入量，而估计器的输出量是原系统状态变量的估计值 $\hat{x}$，它应当满足：

$$\lim_{t\to\infty}(x-\hat{x})=0 \qquad \forall u, x(0), \hat{x}(0) \tag{10-22}$$

根据如图 10.6 所示的关系，可写出观测器部分的状态方程为：

$$\dot{\hat{x}}=\boldsymbol{A}\hat{x}+\boldsymbol{B}u+\boldsymbol{H}(y-\boldsymbol{C}\hat{x})=(\boldsymbol{A}-\boldsymbol{HC})\ \hat{x}+\boldsymbol{B}u+\boldsymbol{H}y \tag{10-23}$$

式（10-23）也可写成：

$$\hat{x}=(\boldsymbol{A}-\boldsymbol{HC})\hat{x}+(\boldsymbol{B}\quad \boldsymbol{H})\begin{pmatrix}u\\ y\end{pmatrix} \tag{10-24}$$

在一类工程实际问题中，产生状态估计值的目的是用以构成反馈控制规律，在这种情况下，完全可以直接讨论如何产生状态线性组合的估计值，而没有必要去产生状态的估计值。

根据观测的状态变量个数，状态观测器可分为全维状态观测器和降维状态观测器。全维状态观测器用来观测全部状态变量，而降维状态观测器只需估计不可测量或没有测量的状态变量。

10.3.1.3　状态观测器的设计原则和设计步骤

由上述定义，不难得出构成系统观测器的原则是：

（1）观测器 $\hat{\Sigma}$ 以原系统 Σ_0 的输入和输出作为其输入。

（2）为了满足等价性指标，原系统 $\hat{\Sigma}$ 应当是完全能观测的，或者 X 中不能观测的部分是渐进稳定的。

（3）$\hat{\Sigma}$ 的输出 $\hat{x}(t)$ 应有足够快的速度逼近，这就要求 $\hat{\Sigma}$ 有足够宽的频带。

（4）$\hat{\Sigma}$ 应有较高的抗干扰性，这就要求 $\hat{\Sigma}$ 有较窄的频带，显然，观测器的快速性和抗干扰性是矛盾的，只能折中地加以选择。

（5）$\hat{\Sigma}$ 在结构上应尽可能简单，即具有尽可能低的维数。

状态观测器的设计步骤如下：

（1）判断系统的可控性，只有系统是可控的，设计状态观测器才有意义。

（2）判断系统的可观性，只有系统是可观的，才能从系统的测量信号估计状态。

（3）确定系统的极点，若根据控制要求给定了极点，则分别确定状态反馈增益矩阵 $\boldsymbol{F}$ 和观测器增益矩阵 $\boldsymbol{K}$。

（4）设计数字补偿器，系统中的估计状态和输出序列两部分均由计算机实现，将两部分的功能组合成一个数字补偿器算式，以便于在计算机上实现。

10.3.2　全维状态观测器

全维状态观测器是利用输出测量值和输入控制值观测系统的全部状态。将原系统输出量 $y(k)$ 和观测器输出量 $\hat{y}(k)$ 之间的误差反馈至状态观测器，构成闭环状态观测器，其观测状态包含了受控系统的全部状态变量。

10.3.2.1　全维状态观测器

考虑如下线性定常系统：

$$\dot{x} = Ax + Bu \tag{10-25}$$

$$y = Cx \tag{10-26}$$

假设状态向量 x 由如下动态方程中的状态 $\tilde{x}$ 来近似，该式表示状态观测器。

$$\dot{\tilde{x}} = A\tilde{x} + Bu + K_e(y - C\tilde{x}) \tag{10-27}$$

注意到状态观测器的输入为 y 和 u，输出为 $\tilde{x}$ 。

式（10-27）的右端最后一项包含被观测输出 $C\tilde{x}$ 之间差的修正项，矩阵 K_e 起到加权矩阵的作用，$\tilde{x}$ 为修正项监控状态变量。当此模型使用的矩阵 A 和 B 与实际系统使用的矩阵 A 和 B 之间存在差异时，由于动态模型和实际系统之间的差异，该附加的修正项将减小这些影响。图 10.7 为系统和全维状态观测器结构示意图。

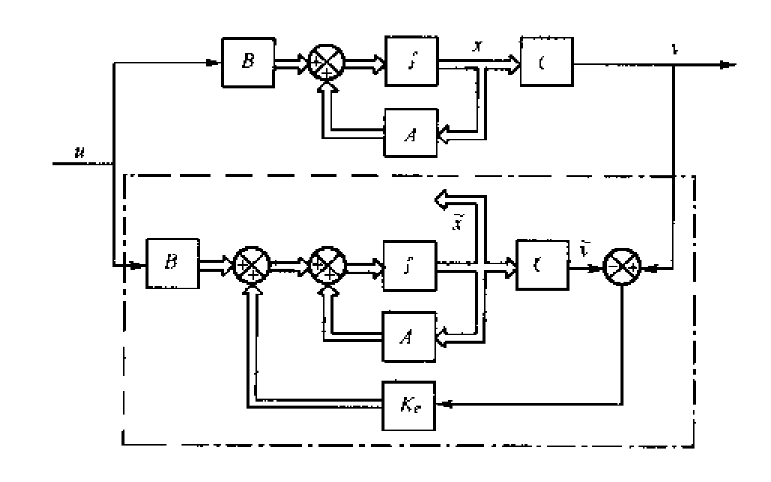


图 10.7　全维状态观测器结构图

式（10-25）减去式（10-27）可得观测器的误差方程为：

$$\dot{x} - \dot{\tilde{x}} = Ax - A\tilde{x} - K_e(Cx - C\tilde{x}) = (A - K_eC)(x - \tilde{x}) \tag{10-28}$$

定义 x 和 $\tilde{x}$ 之差为误差向量，即：

$$e = x - \tilde{x} \tag{10-29}$$

则式（10-28）可改写为：

$$\dot{e} = (A - K_eC)e \tag{10-30}$$

由式（10-30）可看出，误差向量的动态特性由矩阵 $A-K_eC$ 的特征值决定。如果矩阵 $A-K_eC$ 是稳定矩阵，则对任意初始误差向量 $e(0)$，误差向量都将趋近于零。也就是说，不管 $x(0)$和 $\tilde{x}(0)$值如何，$\tilde{x}(t)$ 都将收敛到 $x(t)$。如果所选的矩阵 $A-K_eC$ 的特征值使得误差向量的动态特性渐近稳定且足够快，则任意误差向量都将以足够快的速度趋近于零（原点）。

如果系统是完全能观测的，则证明可以选择 K_e 使得 $A-K_eC$ 具有任意所期望的特征值。也就是说，可以确定观测器的增益矩阵 K_e，以产生所期望的矩阵 $A-K_eC$。

确定观测器增益矩阵 K_e 的问题，也就是全维状态观测器的设计问题，以使由式

（10-30）定义的误差动态方程以足够快的响应速度渐近稳定（渐近稳定性和误差动态方程的响应速度由矩阵 $A-K_eC$ 的特征值决定）。因此，全维观测器的设计就变成为了确定一个合适的 K_e，使得 $A-K_eC$ 具有所期望的特征值，这样，全维状态观测器的设计问题也就变成为了前面所讨论的极点配置问题。

考虑如下的线性定常系统：

$$\begin{aligned}\dot{x}&=Ax+Bu\\ y&=Cx\end{aligned} \tag{10-31}$$

在设计全维状态观测器时，可以求解其对偶问题。也就是说，求解如下对偶系统的极点配置问题：

$$\begin{aligned}\dot{z}&=A^{\mathrm{T}}z+C^{\mathrm{T}}\upsilon\\ n&=B^{\mathrm{T}}z\end{aligned} \tag{10-32}$$

假设控制输入为：

$$\upsilon=-Kz \tag{10-33}$$

如果对偶系统是状态完全能控的，则可确定状态反馈增益矩阵 K，使得矩阵 $A^{\mathrm{T}}-C^{\mathrm{T}}K$ 得到一组期望的特征值。

如果 μ_1，μ_2，…，μ_n 是期望的状态观测器矩阵特征值，则通过取相同的 μ_i 作为对偶系统状态反馈增益矩阵的期望特征值，可得：

$$\left|sI-(A^{\mathrm{T}}-C^{\mathrm{T}}K)\right|=(s-\mu_1)(s-\mu_2)\cdots(s-\mu_n) \tag{10-34}$$

注意到 $A^{\mathrm{T}}-C^{\mathrm{T}}K$ 和 $A-K^{\mathrm{T}}C$ 的特征值相同，可得：

$$\left|sI-(A^{\mathrm{T}}-C^{\mathrm{T}}K)\right|=\left|sI-(A-K^{\mathrm{T}}C)\right| \tag{10-35}$$

比较特征多项式 $\left|sI-(A-K^{\mathrm{T}}C)\right|$ 和观测器系统的特征多项式 $\left|sI-(A-K_eC)\right|$，可找出 K_e 和 K^T 的关系为：

$$K_e=K^{\mathrm{T}} \tag{10-36}$$

因此，通过在对偶系统中由极点配置方法确定矩阵 K，则由关系式 $K_e=K^{\mathrm{T}}$ 就可确定原系统的观测器增益矩阵 K。

如前所述，确定 $A-K_eC$ 所对应的观测器增益矩阵 K_e 的充分必要条件是：原系统的对偶系统

$$\dot{z}=A^*z+C^*v \tag{10-37}$$

是状态完全能控的。该对偶系统状态完全能控的条件是：

$$[\ C^*\ \vdots\ A^*C^*\ \vdots\ \cdots\ \vdots\ (A^*)^{n-1}C^*\ ] \tag{10-38}$$

的秩为 n。这是式（10-25）和式（10-26）所定义的原系统完全能观的条件，即由式（10-25）和式（10-26）所定义的系统状态能观的充分必要条件是系统完全能观。

10.3.2.2　全维状态观测器的设计

考虑由下式定义的线性定常系统：

$$\begin{aligned}\dot{x}&=Ax+Bu\\ y&=Cx\end{aligned} \tag{10-39}$$

式中，$x \in \boldsymbol{R}^n, u \in \boldsymbol{R}^1, y \in \boldsymbol{R}^1, \boldsymbol{A} \in \boldsymbol{R}^{n\times n}, \boldsymbol{B} \in \boldsymbol{R}^{n\times 1}, \boldsymbol{C} \in \boldsymbol{R}^{1\times n}$。假设系统是完全能观测的，又设系统结构如图 10.7 所示。

在设计全维状态观测器时，如果将式（10-39）给出的系统变换为能观测标准形就很方便了。如前所述，可按下列步骤进行：

定义一个变换矩阵 $\boldsymbol{P}$，使得：

$$\boldsymbol{P} = (\boldsymbol{WR})^{-1} \tag{10-40}$$

式中，$\boldsymbol{R}$ 是能观测性矩阵。

$$\boldsymbol{R}^{\mathrm{T}} = [\ \boldsymbol{C}^{\mathrm{T}} \ \vdots\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}} \ \vdots\ \cdots \ \vdots\ (\boldsymbol{A}^{\mathrm{T}})^{n-1}\boldsymbol{C}^{\mathrm{T}}\ ] \tag{10-41}$$

且对称矩阵 $\boldsymbol{W}$ 为：

$$\boldsymbol{W} = \begin{bmatrix} a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\ a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ a_1 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

式中，a_i 是由式（10-39）给出的如下特征方程的系数：

$$|s\boldsymbol{I} - \boldsymbol{A}| = s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n = 0 \tag{10-42}$$

显然，由于假设系统是完全能观测的，因此矩阵 $\boldsymbol{WR}$ 的逆存在。

现定义一个新的 n 维状态向量 ξ：

$$x = \boldsymbol{P}\,\xi \tag{10-43}$$

则式（10-39）为：

$$\begin{cases} \dot{\xi} = \boldsymbol{P}^{-1}\boldsymbol{AP}\xi + \boldsymbol{P}^{-1}\boldsymbol{B}u \\ y = \boldsymbol{CP}\xi \end{cases} \tag{10-44}$$

式中，

$$\boldsymbol{P}^{-1}\boldsymbol{AP} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -a_n \\ 1 & 0 & \cdots & 0 & -a_{n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_1 \end{bmatrix} \tag{10-45}$$

$$\boldsymbol{P}^{-1}\boldsymbol{B} = \begin{bmatrix} b_n - a_n b_o \\ b_{n-1} - a_{n-1} b_o \\ \vdots \\ b_1 - a_1 b_o \end{bmatrix} \tag{10-46}$$

$$\boldsymbol{CP} = [0\ \ 0\ \ \cdots\ \ 0\ \ 1] \tag{10-47}$$

式（10-44）是能观测标准形，因而给定一个状态方程和输出方程，如果系统是完全能观测的，并且通过采用式（10-43）所给出的变换，将原系统的状态向量 x 变换为新的状态向量 ξ，则可将给定的状态方程和输出方程变换为能观测标准形。注意，如果矩阵 $\boldsymbol{A}$ 已经是能观测标准形，则 $\boldsymbol{Q} = \boldsymbol{I}$。

如前所述，选择由

$$\begin{aligned}\dot{\tilde{x}} &= A\tilde{x} + Bu + K_e(y - C\tilde{x}) \\ &= (A - K_eC)\tilde{x} + Bu + K_eCx\end{aligned} \tag{10-48}$$

给出的状态观测器的动态方程。

现定义：

$$\tilde{x} = P\tilde{\xi} \tag{10-49}$$

将式（10-49）代入式（10-48），有：

$$\dot{\tilde{\xi}} = P^{-1}(A - K_eC)P\tilde{\xi} + P^{-1}Bu + P^{-1}K_eCP\xi \tag{10-50}$$

用$\dot{\xi}$减去$\dot{\tilde{\xi}}$，可得：

$$\dot{\xi} - \dot{\tilde{\xi}} = P^{-1}(A - K_eC)P(\xi - \tilde{\xi}) \tag{10-51}$$

定义：

$$\varepsilon = \xi - \tilde{\xi} \tag{10-52}$$

则式（10-52）为：

$$\dot{\varepsilon} = P^{-1}(A - K_eC)P\varepsilon \tag{10-53}$$

要求误差动态方程是渐近稳定的，且$\varepsilon(t)$以足够快的速度趋于零。

确定矩阵 K_e 的步骤是：选择所期望的观测器极点（$A - K_eC$ 的特征值），然后确定 K_e，使其给出所期望的观测器极点。注意 $P^{-1} = WR$，可得：

$$P^{-1}K_e = \begin{bmatrix} a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\ a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ a_1 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-2} \\ CA^{n-1} \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_{n-1} \\ k_n \end{bmatrix} \tag{10-54}$$

式中：

$$K_e = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_n \end{bmatrix} \tag{10-55}$$

由于 $P^{-1}K_e$ 是一个 n 维向量，则

$$P^{-1}K_e = \begin{bmatrix} \delta_n \\ \delta_{n-1} \\ \vdots \\ \delta_1 \end{bmatrix} \tag{10-56}$$

那么 $P^{-1}K_eCP$ 为：

$$P^{-1}K_eCP=\begin{bmatrix}\delta_n\\ \delta_{n-1}\\ \vdots\\ \delta_1\end{bmatrix}[0\ \ 0\ \cdots\ 1]=\begin{bmatrix}0 & 0 & \cdots & 0 & \delta_n\\ 0 & 0 & \cdots & 0 & \delta_{n-1}\\ \vdots & \vdots & & \vdots & \vdots\\ 0 & 0 & \cdots & 0 & \delta_1\end{bmatrix} \tag{10-57}$$

$$P^{-1}(A-K_eC)P=P^{-1}AP-P^{-1}K_eCP=\begin{bmatrix}0 & 0 & \cdots & 0 & -a_n-\delta_n\\ 1 & 0 & \cdots & 0 & -a_{n-1}-\delta_{n-1}\\ 0 & 1 & \cdots & 0 & -a_{n-2}-\delta_{n-2}\\ \vdots & \vdots & & \vdots & \vdots\\ 0 & 0 & \cdots & 1 & -a_1-\delta_1\end{bmatrix}$$

（10-58）

特征方程为：

$$\left| sI-P^{-1}(A-K_eC)P \right|=0 \tag{10-59}$$

即

$$\begin{vmatrix}s & 0 & 0 & \cdots & 0 & a_n+\delta_n\\ -1 & s & 0 & \cdots & 0 & a_{n-1}+\delta_{n-1}\\ 0 & -1 & s & \cdots & 0 & a_{n-2}+\delta_{n-2}\\ \vdots & \vdots & \vdots & & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & -1 & s+a_1+\delta_1\end{vmatrix}=0 \tag{10-60}$$

或者

$$s^n+(a_1+\delta_1)s^{n-1}+(a_2+\delta_2)s^{n-2}+\cdots+(a_n+\delta_n)=0 \tag{10-61}$$

可见，$\delta_n, \delta_{n-1},\cdots, \delta_1$ 中的每一个只与特征方程系数中的一个相关联。

假设误差动态方程所期望的特征方程为：

$$(s-\mu_1)(s-\mu_2)\cdots(s-\mu_n)=s^n+a_1^*s^{n-1}+a_2^*s^{n-2}+\cdots+a_{n-1}^*s+a_n^*=0 \tag{10-62}$$

注意，期望的特征值μ_i 确定了被观测状态以多快的速度收敛于系统的真实状态。比较式（10-61）和式（10-62）的 s 同幂项的系数，可得：

$$\begin{aligned}a_1+\delta_1&=a_1^*\\ a_2+\delta_2&=a_2^*\\ &\cdots\\ a_n+\delta_n&=a_n^*\end{aligned} \tag{10-63}$$

从而有：

$$\begin{aligned}\delta_1&=a_1^*-a_1\\ \delta_2&=a_2^*-a_2\\ &\cdots\\ \delta_n&=a_n^*-a_n\end{aligned} \tag{10-64}$$

于是，由式（10-56）得到：

$$P^{-1}K_e=\begin{bmatrix}\delta_n\\ \delta_{n-1}\\ \vdots\\ \delta_1\end{bmatrix}=\begin{bmatrix}a_n^*-a_n\\ a_{n-1}^*-a_{n-1}\\ \vdots\\ a_1^*-a_1\end{bmatrix} \tag{10-65}$$

因此

$$K_e=P\begin{bmatrix}a_n^*-a_n\\ a_{n-1}^*-a_{n-1}\\ \vdots\\ a_1^*-a_1\end{bmatrix}=(WR)^{-1}\begin{bmatrix}a_n^*-a_n\\ a_{n-1}^*-a_{n-1}\\ \vdots\\ a_1^*-a_1\end{bmatrix} \tag{10-66}$$

式（10-67）规定了所需的状态观测器增益矩阵 K_e。

如前所述，式（10-66）也可通过其对偶式得到。也就是说，考虑对偶系统的极点配置问题，并求出对偶系统的状态反馈增益矩阵 K，那么状态观测器的增益矩阵 K 就可由 K^T 来确定。

一旦选择了所期望的特征值（或所期望的特征方程），只要系统完全能观测，就能设计出全维状态观测器。所选择的特征方程的期望特征值应能使状态观测器的响应速度至少比所考虑的闭环系统快 2～5 倍。

如前所述，全维状态观测器的方程为：

$$\dot{\tilde{x}}=(A-K_eC)\tilde{x}+Bu+Ky \tag{10-67}$$

注意：迄今为止，假设观测器中的矩阵 A 和 B 与实际系统中的完全相同。实际上这是做不到的，因此误差动态方程不可能由式（10-53）给出，这就意味着误差不可能趋于零。因此，应尽可能建立观测器的准确数学模型，以使误差小到令人满意的程度。

下面介绍如何用 MATLAB 设计状态观测器。

【例 10-4】　考虑一个调节器系统的设计。给定线性定常系统为：

$$\dot{x}=Ax+Bu$$
$$y=Cx$$

式中

$$A=\begin{bmatrix}0 & 1\\ 20.6 & 0\end{bmatrix},\ B=\begin{bmatrix}0\\ 1\end{bmatrix},\ C=[1\quad 0]$$

且闭环极点为 $s=\mu_i(i=1,2)$，其中 $\mu_1=-1.8+j2.4$，$\mu_2=-1.8-j2.4$。期望用观测状态反馈控制，而不是用真实的状态反馈控制。观测器的期望特征值为 $\mu_1=\mu_2=-8$。试采用 MATLAB 确定相应的状态反馈增益矩阵 K 和观测器增益矩阵 K_e。

解：MATLAB 的程序代码如下。

```
A=[0  1; 20.6  0];
B=[0; 1]
C=[1  0];
D=[0];
Q=[B, A*B];
```

```
%计算能控矩阵的秩
Rank(Q)
```

运行结果如下：

```
%能控矩阵的秩
ans=
     2
```

由输出结果可知，系统完全可控，因此可以实现任意的极点配置。

接着输入如 MATLAB 程序代码：

```
J=[-1.8+2.4*i 0; 0 -1 8-2 4*i];
%计算期望闭环极点的多项式
Poly(J)
Phi=polyvalm(poly(J),A);
%计算状态反馈增益矩阵 K
K=[0 1]*inv(Q)*Phi
RT=[C', A'*C'];
%计算能观矩阵的秩
rank(RT)
```

计算输出结果如下：

```
%Poly(J)
ans=
    1.0000    3.6000    9.0000
K=
   29.6000    3.6000
%rank(RT)
ans=
     2
```

由输出结果可知，系统完全可观，因此可以对系统设计观测器。

接着输入如下 MATLAB 程序代码：

```
JO=[-8 0; 0 -8];
%计算期望观测器极点的多项式
poly(JO)
Ph=polyvalm(poly(JO),A);
%计算观测器增益矩阵 Ke
Ke=Ph*(inv(RT'))*[0; 1]
```

计算输出结果如下：

```
%poly(JO)
ans=
     1    16    64
```

```
Ke=
    16.0000
    84.6000
```

由输出结果可知，所求状态反馈增益矩阵 $\boldsymbol{K}$ 为：

$$\boldsymbol{K}=[29.6 \quad 3.6]$$

观测器增益矩阵 $\boldsymbol{K}_e$ 为：

$$\boldsymbol{K}_e=\begin{bmatrix}16\\84.6\end{bmatrix}$$

该观测状态反馈控制系统是 4 阶的，其特征方程为：

$$|\ s\boldsymbol{I}-\boldsymbol{A}+\boldsymbol{BK}\ |\ |\ s\boldsymbol{I}-\boldsymbol{A}+\boldsymbol{K}_e\boldsymbol{C}\ |=0$$

将期望的闭环极点和期望的观测器极点代入上式，接着输入如下代码：

```
X=[eig(A-B*K); eig(A-Ke*C)]
%计算特征多项式|sI-A+BK||sI-A+KeC|
poly(X)
```

计算输出结果如下：

```
X=
  -1.8000+2.4000i
  -1.8000-2.4000i
  -8.0000
  -8.0000
% poly(X)
ans=
    1.0000    19.6000    130.6000    374.4000    576.0000
```

由输出结果可知：

$$\begin{aligned}|s\boldsymbol{I}-\boldsymbol{A}+\boldsymbol{BK}\ |\ |\ s\boldsymbol{I}-\boldsymbol{A}+\boldsymbol{K}_e\boldsymbol{C}\ |&=(s+1.8-\mathrm{j}2.4)(s+1.8+\mathrm{j}2.4)(s+8)^2\\&=s^4+19.6s^3+130.6s^2+374.4s+576\end{aligned}$$

10.3.3 降维状态观测器

当状态观测器估计状态向量的维数小于被控对象状态向量的维数时，称为降维状态观测器。对于 q 维输出系统，有 q 个输出变量可直接测量得到，通过线性变换，有 q 个状态变量可由输出得到，观测器只需估计（$n-q$）个状态变量，称为（$n-q$）维状态观测器。它是一个（$n-q$）维子系统。

（$n-q$）维子系统动态方程的建立步骤如下所述。

设 n 维受控系统的动态方程为：

$$\begin{cases}\dot{x}=\boldsymbol{A}x+\boldsymbol{B}u\\y=\boldsymbol{C}x\end{cases}\tag{10-68}$$

系统有 q 个输出，且系统是可控和可观测的。系统可控保证了可以任意配置系统的极点，系统可观测确定了降维状态观测器的维数为（$n-q$）。把 x 分解为 $\bar{x}_1$ 和 $\bar{x}_2$（q 个可直接由输出测得的状态变量）两部分。

引入非奇异线性变换：

$$x = T\bar{x} \tag{10-69}$$

其中，

$$T_{n\times n} = \begin{bmatrix} D \\ \cdots \\ C \end{bmatrix} \begin{matrix} (n-q) \\ \\ q \end{matrix} \tag{10-70}$$

变换后的系统动态方程为：

$$\begin{cases} \dot{\bar{x}} = \bar{A}\bar{x} + \bar{B}u \\ \bar{y} = y = \bar{C}\bar{x} \end{cases} \tag{10-71}$$

其中，

$$\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix}$$

$$\bar{A} = T^{-1}AT = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \begin{matrix} n-q \\ q \end{matrix}$$

$$\bar{B} = T^{-1}B = \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \end{bmatrix} \begin{matrix} n-q \\ q \end{matrix}$$

$$\bar{C} = CT = C\begin{bmatrix} D \\ C \end{bmatrix}^{-1}$$

$$\begin{cases} C = CT = C\begin{bmatrix} D \\ C \end{bmatrix}^{-1}\begin{bmatrix} D \\ C \end{bmatrix} = \bar{C}\begin{bmatrix} D \\ C \end{bmatrix} \\ C = \begin{bmatrix} 0 & I \end{bmatrix}\begin{bmatrix} D \\ C \end{bmatrix} \end{cases}$$

$$\bar{C} = \underset{n-q\quad q}{\begin{bmatrix} 0 & I \end{bmatrix}}\ q \qquad \bar{y} = \bar{x}_2$$

则变换后的状态方程展开为：

$$\begin{cases} \dot{\bar{x}}_1 = \bar{A}_{11}\bar{x}_1 + \bar{A}_{12}\bar{x}_1 + \bar{B}_1u = \bar{A}_{11}\bar{x}_1 + \bar{A}_{12}\bar{y} + \bar{B}_1u \\ \dot{\bar{y}} = \bar{A}_{21}\bar{x}_1 + \bar{A}_{22}\bar{y} + \bar{B}_2u \end{cases} \tag{10-72}$$

令 $v = \bar{A}_{12}\bar{y} + \bar{B}_1u$，并把它看做（$n-q$）维子系统的输入向量；$z = \dot{y} - \bar{A}_2\bar{y} - \bar{B}_2u$，并把它看做（$n-q$）维子系统的输出向量，则（$n-q$）维子系统的动态方程为：

$$\begin{cases} \dot{\bar{x}}_1 = \bar{A}_{11}\bar{x}_1 + v \\ z = \bar{A}_{21}\bar{x}_1 \end{cases} \tag{10-73}$$

由于原 n 维系统可观测，因此该（$n-q$）维子系统也是可观测的。

与全维状态观测器的构成方法相同，设计（$n-q$）维子系统的全维状态观测器，其原理结构图如图 10.8 所示。

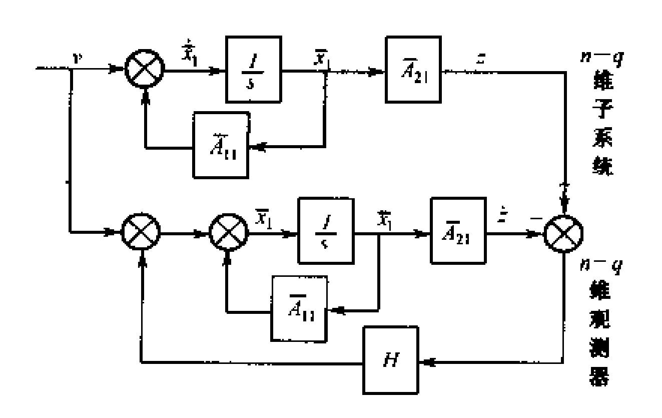


图 10.8　（n-q）维子系统及其观测器原理结构图

降维状态观测器的动态方程为：

$$\begin{cases}\dot{\hat{\bar{x}}}_1=\bar{A}_{11}\hat{\bar{x}}_1+v+H(z-\hat{z})\\ \hat{z}=\bar{A}_{21}\hat{\bar{x}}_1\end{cases}\tag{10-74}$$

代入 v 和 z 的表达式，有：

$$\dot{\hat{\bar{x}}}_1=(\bar{A}_{11}-H\bar{A}_{21})\hat{\bar{x}}_1+(\bar{A}_{21}\bar{y}+\bar{B}_1u)+H(\dot{\bar{y}}-\bar{A}_{22}\bar{y}-\bar{B}_2u)\tag{10-75}$$

由于式中含有导数项 $\dot{\bar{y}}$，将影响估计状态的惟一性，另选状态变量以使状态方程中不含 $\dot{\bar{y}}$，设

$$w=\hat{\bar{x}}_1-H\bar{y}\tag{10-76}$$

则 $\dot{w}=\dot{\hat{\bar{x}}}_1-H\dot{\bar{y}}$，于是

$$\dot{w}=(\bar{A}_{11}-H\bar{A}_{21})w+(\bar{B}_1-H\bar{B}_2)u+\left[(\bar{A}_{11}-H\bar{A}_{21})H+\bar{A}_{12}-H\bar{A}_{22}\right]\bar{y}\tag{10-77}$$

$$\hat{\bar{x}}_1=w+H\bar{y}\tag{10-78}$$

上式即为降维状态观测器的状态方程，式中 $(\bar{A}_{11}-H\bar{A}_{21})$ 为降维观测器状态矩阵，降维观测器的极点由下列特征方程决定：

$$\left|\lambda I-(\bar{A}_{11}-H\bar{A}_{21})\right|=0\tag{10-79}$$

用做状态反馈的估计状态向量由两部分组成：由（$n-q$）维状态观测器给出的状态估值 $\hat{\bar{x}}_1$；由输出传感器测得的状态 $\bar{x}_2=\bar{y}$。

因为 $\hat{\bar{x}}_1=w+H\bar{y}$，所以

$$\hat{\bar{x}}=\begin{bmatrix}\hat{\bar{x}}_1\\ \cdots\\ \bar{y}\end{bmatrix}=\begin{bmatrix}w+H\bar{y}\\ \cdots\\ \bar{y}\end{bmatrix}=\begin{bmatrix}I_{n-q} & H\\ 0 & I_q\end{bmatrix}\begin{bmatrix}w\\ \cdots\\ \bar{y}\end{bmatrix}\tag{10-80}$$

于是 $\hat{x}=T\hat{\bar{x}}$。

带降维观测器的全状态反馈系统的设计步骤可归纳如下：

（1）检查受控系统的可观测性，确定降维观测器的维数（$n-q$）；

（2）运用非奇异线性变换 $x = T\bar{x}$，将可由传感器测得的 q 个状态变量与待观测器估计的（$n-q$）个状态变量分离；

（3）构造（$n-q$）维状态观测器，观测器反馈系数矩阵 H 由观测器期望特征值求得；

（4）将 $\bar{x}$ 变换回到原系统状态空间，将估值 $\hat{x}$ 作为原系统状态反馈的状态信息。

按上述方法构成的（$n-q$）维观测器称为龙博格观测器。在变换的状态空间内的龙博格观测器结构如图 10.9 所示。

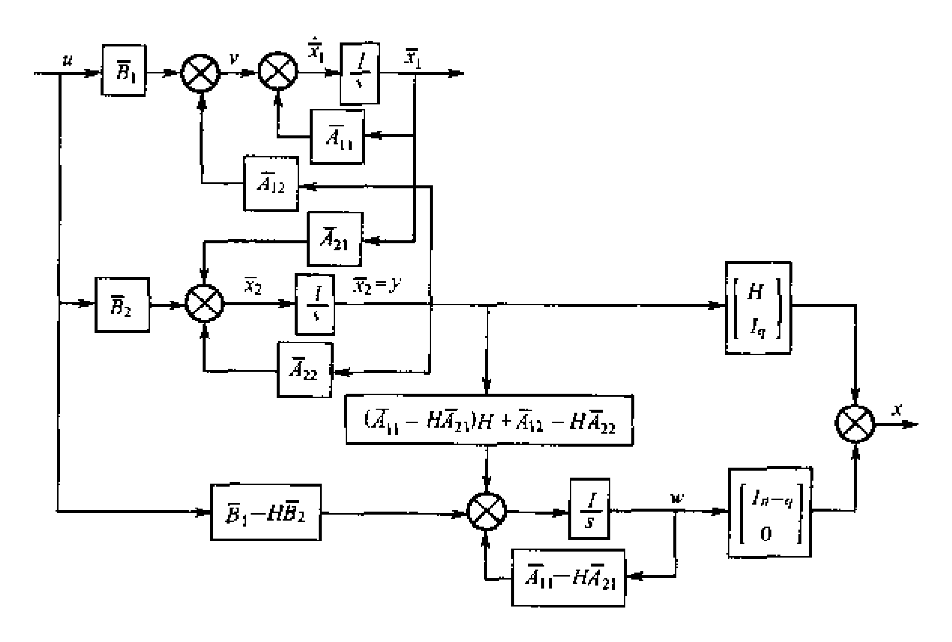


图 10.9　龙博格观测器结构图

【例 10-5】　某给定线性定常系统为：

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

式中，

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix},\quad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix},\quad C = [1 \ \cdots \ 0 \ \ 0]$$

假定状态变量 x_1（等于 y）是可量测的，但未必是能观测的，试确定最小阶观测器的增益矩阵 K_e。期望的特征值为：

$$\mu_1 = -2 + \mathrm{j}2\sqrt{3},\ \mu_2 = -2 - \mathrm{j}2\sqrt{3}$$

试分别利用变换矩阵 P 方法和爱克曼公式法来实现。

解：

（1）利用变换矩阵 P 方法，MATLAB 程序代码如下：

```
A=[0  1  0;0  0  1;-6  -11  -6];
B=[0; 0; 1];
```

```
Aaa=[0];
Aab=[1  0];
Aba=[0; -6];
Abb=[0  1; -11 -6];
Ba=[0];
Bb=[0; 1];
P=poly(Abb)
```

运行结果如下：

```
P=
    1.0000    6.0000    11.0000
```

接着输入如下 MATLAB 程序代码：

```
a1=P(2);
a2=P(3);
RT=[Aab'  Abb'*Aab'];
W=[a1  1; 1  0];
J=[-2+2*sqrt(3)*i  0; 0  -2-2*sqrt(3)*i];
JJ=poly(J)
```

运行结果如下：

```
JJ=
    1.0000    4.0000    16.0000
```

接着输入如下 MATLAB 程序代码：

```
aa1=JJ(2);
aa2=JJ(3);
Ke=inv(W*RT')*[aa2-a2; aa1-a1]
```

运行结果如下：

```
Ke=
  -2.0000
  17.0000
```

观测器增益矩阵 $\boldsymbol{K}_e$ 为：

$$\boldsymbol{K}_e=\begin{bmatrix}-2\\17\end{bmatrix}$$

（2）利用爱克曼公式法，MATLAB 程序代码如下：

```
A=[0  1  0; 0  0  1; -6  -11  -6];
B=[0; 0; 1];
Aaa=[0];
Aab=[1  0];
```

```
Aba=[0; -6];
Abb=[0  1; -11  -6];
Ba=[0];
Bb=[0; 1];
RT=[Aab'  Abb'*Aab'];
J=[-2+2*sqrt(3)*i  0; 0  -2-2*sqrt(3)*i];
JJ=poly(J)
```

运行结果如下：

```
JJ=
    1.0000    4.0000    16.0000
```

接着输入如下 MATLAB 程序代码：

```
Phi=polyvalm(poly(J),Abb);
Ke=Phi*inv(RT')*[0; 1]
```

运行结果如下：

```
Ke=
    -2
    17
```

由运算结果可知，观测器增益矩阵 $\boldsymbol{K}_e$ 为：

$$\boldsymbol{K}_e=\begin{bmatrix}-2\\17\end{bmatrix}$$

从运算结果可以看出，利用变换矩阵 P 方法和爱克曼公式法的结果完全相同。

第 11 章　非线性系统

11.1　引　　言

物理系统具有固有非线性，所有控制系统都具有一定程度的非线性。现代技术对控制系统提出了更为严格的要求，因而非线性系统及其控制引起越来越多的重视，对非线性系统的研究已成为控制工程领域的一项重要内容。

对某些控制系统，为了简化分析过程，可通过在工作点附近线性化来处理，但当系统中包含有本质性的非线性特性时，就不能用线性化的方法来简化处理。非线性系统与线性系统有本质的差别，非线性系统不满足叠加原理，其稳定性不仅取决于控制系统的固有结构和参数，而且与系统的初始条件和输入信号有关。

通过本章，读者能了解非线性系统的发展概况、非线性系统的数学描述和特性、非线性系统的研究方法和特点，掌握非线性系统分析和设计的基本概念和方法以及利用 MATLAB/Simulink 对非线性系统进行分析。

11.2　非线性系统概述

11.2.1　非线性控制理论发展概况

在控制领域，人们最先研究的控制系统都是线性的。对线性系统的物理描述和数学求解是比较容易实现的事情，而且已经形成了一套完善的线性理论和分析研究方法。但对非线性系统来说，除极少数情况外，目前还没有一套可行的通用方法，而且每种方法只概适用于某一类问题，不能普遍适用。

目前对非线性控制系统的认识和处理基本上还是处于初级阶段。另一方面，从对控制系统的精度要求来看，用线性系统理论来处理目前绝大多数工程技术问题在一定范围内都可以得到比较满意的结果，因此，一个物理系统的非线性因素常常被忽略了，或者被各种线性关系所简化或代替。这就是线性系统理论发展迅速并趋于完特，而非线性系统理论长期得不到重视和发展的主要原因。

随着科学技术的不断发概，人们对实际生产过概的分析要求日益精密，各种较为精确的分析和科学实验结果表明，任何一个实际的物理系统都是非线性的。所谓线性只是对非线性的一种简化或近似，或者说是非线性的一种特例。

到 20 世纪 40 年代，对非线性控制系统的研究已取得一些明显的进概。目前主要的非线性分析方法有：相平面法、李亚普诺夫法和描述函数法等。这些方法都已经被广泛用来解决实际的非线性系统问概。

但是这些方法都有一定的局限性，都不能成为分析非线性系统的通用方法。例如，用相平面法虽然能够获得系统的全部特征，如稳定性、过渡过程等，但对于大于三阶的系统无法应用。李亚普诺夫法则仅限于分析系统的绝对稳定性问题，而且要求非线性元件的特性满足一定条件。虽然这些年来，国内外有不少学者一直在这方面进行探讨，也研究出了一些新的方法，如频域的波波夫判据、广义圆判据、输入/输出稳定性理论等，但总的来说，非线性控制系统理论目前仍处于发展阶段，远非线性系统那样完善，很多问题都还有待解决，未来的研究领域十分宽广。

非线性控制理论具有广阔的发展前景，将成为本世纪控制理论的主旋律，并为人类社会提供更加先进的控制系统，使自动化水平取得更大进步。

11.2.2　典型非线性特性

含有非线性元件或环节的系统称为非线性系统。非线性特性包括许多类型，典型的静态非线性特性包括死区非线性、饱和非线性、间隙非线性和继电非线性，下面分别加以介绍。

11.2.2.1　死区非线性

死区非线性是一种常见的非线性，如图 11.1 所示，它通常是由放大器、传感器、执行机构的不灵敏区造成的。

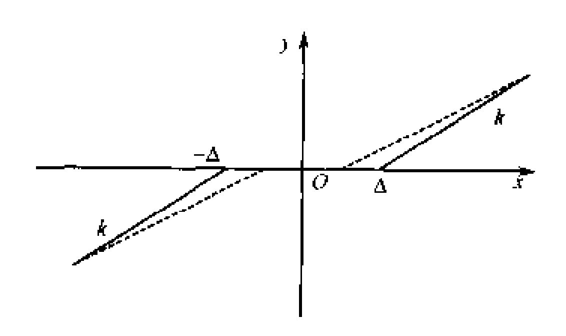


图 11.1　死区非线性

理想的死区非线性一般可用图 11.1 中的三段直线（实线）来表示；实际的死区非线性一般可用图 11.1 中的点划线来表示。理想的死区非线性可用数学模型描述为：

$$y=\begin{cases}k(x+\Delta) & x<-\Delta\\ 0 & |x|\leqslant\Delta\\ k(x-\Delta) & x>\Delta\end{cases}\tag{11-1}$$

从式（11-1）可以看出，当输入$|x|<\Delta$时，输出为 0，系统处于开环状态，失去调节作用，控制灵敏度下降，稳态误差加大，给控制系统带来不利影响；另一方面，当系统输入端存在小扰动信号时，死区可减小扰动信号，给控制系统带来有利影响，有些系统为提高扰干扰能力而引入死区非线性。就静态特性而言，死区非线性总是增大系统稳态误差。

11.2.2.2　饱和非线性

饱和非线性如图 11.2 所示，它是由饱和引起的。当输入增大到某个值以后，输出便

不再变化，这种现象称为饱和。任何实际的控制系统都存在饱和非线性，因为其输出不可能无限增大，这是放大元件或执行元件的固有特性。

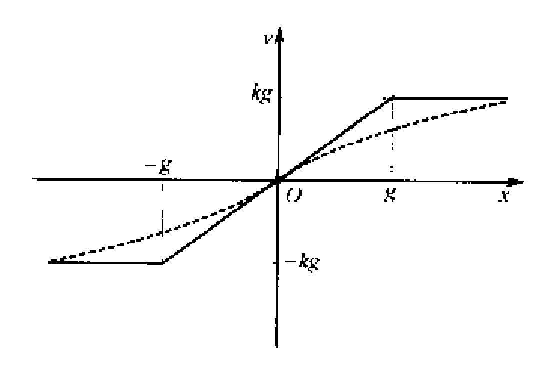


图 11.2　饱和非线性

理想的饱和非线性一般可用图 11.2 中的三段直线来表示：实际的饱和非线性一般可用图 11.2 中的点划线表示。理想的饱和非线性特性可用数学模型描述为：

$$y=\begin{cases}-kg & x<-g \\ kx & |x|\leqslant g \\ kg & x>g\end{cases} \tag{11-2}$$

饱和非线性往往促使系统稳定，但会减小放大系数，因而降低稳态精度。有时出于技术上的要求，采用限幅，以使特性的线性区变窄，实际上是利用了饱和非线性。

11.2.2.3　间隙非线性

间隙非线性如图 11.3 所示，其形成原因通常是由于滞后的作用，类似于线性系统的滞后环节，但不完全等价。齿轮传动是典型的间隙非线性。

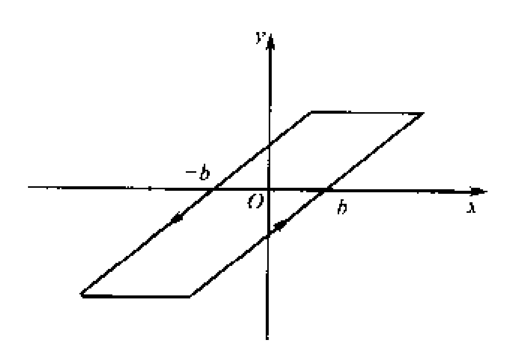


图 11.3　间隙非统性

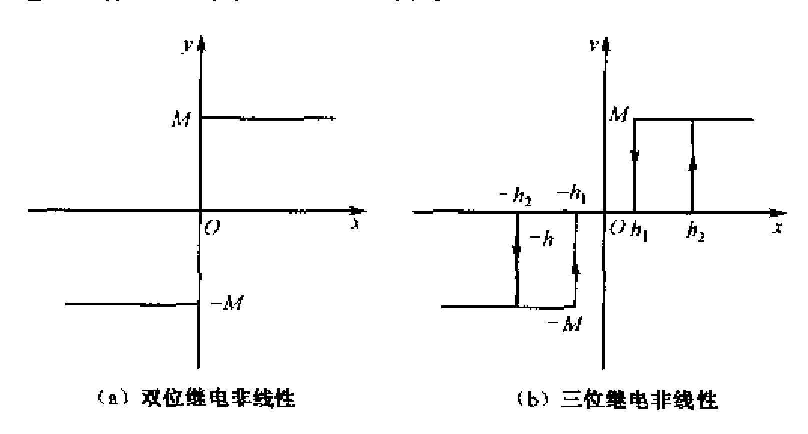
间隙非线性可用数学模型描述为：

$$y=\begin{cases}k(x-b), & x-y/k=b \\ 0, & -b<x-y/k<b \\ k(x+b), & x-y/k=-b\end{cases} \tag{11-3}$$

间隙非线性对控制总是有害的，它会引起系统不稳定或自振滞，对系统稳定品质也不利，故应消除或削弱它的影响。

11.2.2.4　继电非线性

继电非线性顾名思义就是继电器所具有的非线性，其他装置（如电磁阀、斯密特触发器等）都具有类似的非线性特性。它常见的种类有双位继电非线性，如图 11.4（a）所示，以及三位继电非线性，如图 11.4（b）所示。



（a）双位继电非线性　　（b）三位继电非线性

图 11.4　继电非线性

双位继电非线性可用数学模型描述为：

$$y=\begin{cases}+M & x>0\\ -M & x<0\end{cases} \tag{11-4}$$

三位继电非线性（b）可用数学模型描述为：

$$\begin{aligned}&\dot{x}>0:\ y=\begin{cases}+M & x>h\\ 0 & h>x>-mh\\ -M & x<-mh\end{cases}\\ &\dot{x}<0:\ y=\begin{cases}+M & x>mh\\ 0 & mh>x>-h\\ -M & x<-h\end{cases}\end{aligned} \tag{11-5}$$

继电非线性常会使系统产生振荡，对系统是不利的。分析继电非线性具有十分重要的意义，因为采用继电器、电磁阀等元件的控制系统比比皆是，例如大多数家用电冰箱、空调就属于继电器控制系统，研究继电非线性的目的就是为了克服其不利影响，最终使具有继电非线性的控制系统更加完善。

11.3　相 平 面 法

相平面法是庞加莱（Poincare）于 1885 年首先提出的，它通过图解法把求解一阶和二阶系统运动方程组转化成位置和速度平面上的相轨迹，能直观、准确地反映系统的稳定性、平衡状态和稳态精度以及初始条件和参数对系统运动的影响。

应用相平面法分析一阶尤其是二阶非线性控制系统，弄清非线性系统的稳定性、稳定域等基本属性以及解释极限环等特殊现象，具有非常直观形象的效果。

由于绘制二维以上的相轨迹十分困难，因此相平面法对于二阶以上的系统几乎无能为力，这是相平面法的局限。

11.3.1　相平面法基础知识

11.3.1.1　相轨迹和相平面

对于二阶时不变系统可用式（11-6）所示的常微分方程描述：

$$\ddot{x}=f(x,\dot{x}) \tag{11-6}$$

式中，$f(x,\dot{x})$ 是 $x(t)$ 和 $\dot{x}(t)$ 的线性或非线性函数。该方程的解可以用 $x(t)$ 的时间函数曲线表示。

把式（11-6）所示的二阶微分方程改写成二元一阶微分方程组：

$$\begin{cases}\dfrac{\mathrm{d}x}{\mathrm{d}t}=\dot{x}\\ \dfrac{\mathrm{d}\dot{x}}{\mathrm{d}t}=f(x,\dot{x})\end{cases} \tag{11-7}$$

式中，x 可以看做广义位移，$\dot{x}$ 可以看做广义速度。

对式（11-7）所示的微分方程组求解，可以得到解 $x(t)$ 和 $\dot{x}(t)$。

如果取 $x(t)$ 和 $\dot{x}(t)$ 为坐标，以时间 t 为参变量，就可以用 $x(t)$ 和 $\dot{x}(t)$ 的关系曲线表示方程的解，而 t 为参变量。此时，系统每一时刻的状态均对应于该平面上的一点，当参变量 t 变化时，方程的解在 $x-\dot{x}$ 平面上绘出的曲线即表征了系统的运动过程，这个曲线就是相轨迹。相轨迹上箭头符号表示参变量时间 t 的增加方向。两个变量 $x(t)$ 和 $\dot{x}(t)$ 构成的直角坐标系称为相平面。

根据微分方程解的存在与惟一性定理，对于任一给定的初始条件，相平面上有一条相轨迹与之对应。多个初始条件下的运动对应多条相轨迹，形成相轨迹簇，而由一簇相轨迹所组成的图形称为相平面图。

11.3.1.2　奇点

对于二阶系统 $\ddot{x}=f(x,\dot{x})$，相平面上满足 $\dot{x}=0$ 且 $\ddot{x}=0$ 的点叫做奇点，记为 X_e。奇点坐标 $(x_e,\dot{x}_e)$ 是下列代数方程的解，显然奇点一定在坐标轴上。

$$\begin{cases}\dot{x}=0\\ f(x,\dot{x})=0\end{cases} \tag{11-8}$$

对于二阶系统，$\dot{x}=0$ 和 $\ddot{x}=0$ 表示系统的“速度”和“加速度”均为零，也就意味着系统不再运动，因此，奇点又称平衡点。相平面上任何其他点都叫普通点。奇点又分稳定奇点和不稳定奇点。

11.3.1.3　相轨迹切线斜率

由式（11-7）可知，相轨迹上任一点的切线斜率为：

$$\alpha(x,\dot{x})=\frac{\mathrm{d}\dot{x}}{\mathrm{d}x}=\frac{f(x,\dot{x})}{\dot{x}} \tag{11-9}$$

某点的切线斜率就是相轨迹通过该点的运动方向。

在奇点处，$\alpha(x,\dot{x})=\frac{\mathrm{d}\dot{x}}{\mathrm{d}x}=\frac{0}{0}$，其值不定，表明系统在奇点处可以按任意方向趋近或离开奇点，因此，在奇点处多条相轨迹相交。

等倾线就是相轨迹场上所有切线斜率等于某一常数的点的连线。

11.3.1.4　相轨迹图形特征

如果微分方程（11-7）满足解的存在性和惟一性条件，那么相轨迹图一定有如下基本特征：

（1）相轨迹不相交，即相平面上任一普通点有且只有一条相轨迹通过（坐标原点除外）；

（2）相轨迹必垂直通过坐标轴；

（3）相平面横轴上方的相轨迹从左向右运动，横轴下方的相轨迹从右向左运动。

11.3.1.5　极限环

极限环是非线性系统特有的现象，它对应的响应曲线是等幅振荡，也是一种随处可见的现象，可以说，凡是能持续振荡的动态系统都是运行在稳定极限环上的。钟摆的摆动、电子振荡器等都是例证。显然，在干扰环境中，线性系统不可能产生持续等幅振荡，因为极微小的干扰就可能导致振荡发散或衰减到零。

极微小的干扰就导致系统振荡发散或表减到零的极限环称为不稳定极限环。即使干扰使振荡短时离开极限环，干扰消失后则又回到的极限环称为稳定极限环。

11.3.2　相轨迹图绘制

相轨迹图有多种绘制办法，主要有以下三种：

（1）手工绘制概略图。概略图就像相轨迹的“素描”，它是根据相轨迹的基本特征、特殊点、特殊线等信息而“随手”画出的草图，它虽然在具体细节上缺乏精度，但能提供许多重要的定性结论。

（2）手工绘制近似图。在计算机未得到广泛应用的年代，人们研究出好几种手工近似作图法，如等倾线法、δ 法等。这些手工作图法要绘出有一定精度的相轨迹图是十分烦琐的，如今已没有多大实用价值。

（3）计算机绘制精确图。借助计算机数值解法以及 MATLAB/Simulink 等软件绘制相轨迹图。

下面讲述采用 MATLAB/Simulink 绘制相轨迹图的方法。

11.3.2.1　采用 MATLAB 绘制相轨迹图

绘制相轨迹图的实质是求解微分方程的解。求解微分方程数值解的算法有多种，

MATLAB 提供了求解微分方程的函数组，常用的有 ode45，它采用的计算方法是变步长的龙格—库塔 4/5 阶算法。

ode45( )的调用格式如下：

```
[t, y] = ode45(odefun, tspan, y0)
```

在用户自己编写的 MATLAB 函数中既可以描述线性系统特性，也可以描述非线性系统特性。描述系统模型的文件名可以由字符串变量名 odefun 给出；参数 tspan 可以由初始时间 t0 和终止时间 tfinal 构成向量给出，如 tspan = [t0 tfinal]，参数 y0 为系统状态变量初始值，其默认值是一个空矩阵。函数调用后，将返回系统的时间向量 t 和状态变量 y。

【例 11-1】　已知一个二阶线性系统的微分方程为：

$$\ddot{x}+ax=0,\quad a>0,\quad x(t_0)=x_0\quad,\quad \dot{x}(t_0)=\dot{x}_0$$

其中 $a=2$，试用 MATLAB 函数绘制系统的相平面图和零输入响应曲线。

解：取状态变量 $x_2=x,\quad x_1=\dot{x}$，得到系统状态方程模型，即一阶常微分方程组：

$$\begin{cases}\dot{x}_2=\dfrac{\mathrm{d}x}{\mathrm{d}t}=x_1\\ \dot{x}_1=\ddot{x}=-ax_2\end{cases}$$

由此模型就可以用 MATLAB 来求解了。

主程序 MATLAB 代码如下：

```
% test_11_1 是系统微分方程的描述函数
%初始化状态变量为[0, 1]，计算时间向量为[0, 20]
[t, x]=ode45('test_11_1', [0, 20], [0, 1]);
%初始化状态变量为[1, 1]，计算时间向量为[0, 20]
[t1, x1]=ode45('test_11_1', [0, 20], [1, 1]);
%绘制相轨迹
plot(x(:, 1), x(:, 2), '-', x1(:, 1), x1(:, 2), '.');
xlabel('x')
ylabel('dx/dt')
grid
title('相轨迹图')
```

运行结果如下：

```
%例 11-1 程序的子函数代码
%状态导数
function xdot= test_11_1(t, x)
%导数关系式
xdot=[-2*x(2); x(1)];
%格式
%  function xdot=filename(t, x)
%  xdot=[表达式 1；表达式 2；表达式 3；...；表达式 n-1]
```

```
%   表达式 1  对应   x1'=x2
%   表达式 2  对应   x2'=x3
%   表达式 3  对应   x3'=x4
%   ...
%   表达式 n-1  对应   xn-1'=xn
```

主函数运行后，输出如图 11.5 所示的相轨迹图。

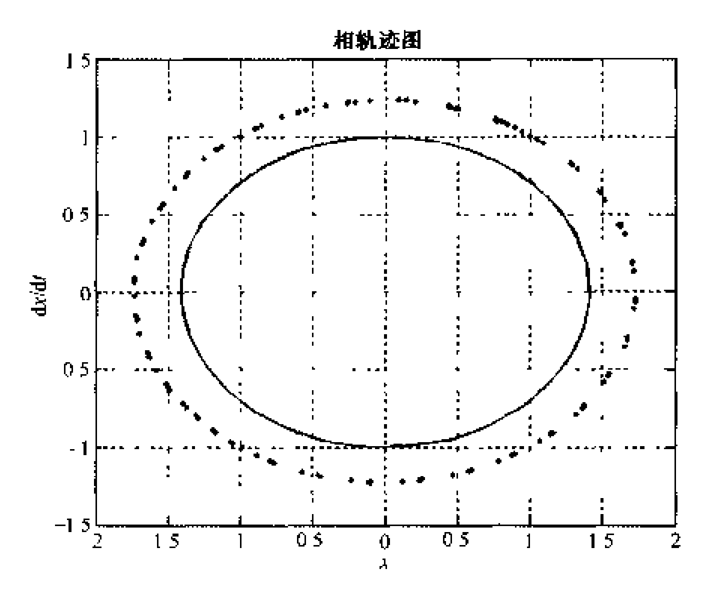


图 11.5　例 11-1 相轨迹图

然后输入如下 MATLAB 程序代码来绘制系统时域响应图：

```
%绘制时域响应曲线
plot(t, x(:, 2), '-', t1, x1(:, 2), '.')
xlabel('t')
ylabel('x(t)')
grid
title('时间响应曲线')
```

运行后，输出如图 11.6 所示，从图中可以看出是等幅振荡。

本题的相轨迹是如图 11.5 所示的椭圆。如果取遍所有的初始值，就会得到无数个一环套一环的椭圆，称为相轨迹场。相轨迹场布满了整个相平面，相轨迹场从全局上展示了动态系统的运动过程，图 11.5 中只绘出了相轨迹场中的 2 根相轨迹。

【例 11-2】　已知一个二阶非线性系统的微分方程为：

$$\ddot{x}+(x^2-1)\dot{x}+x=0,\quad x(0)=3,\quad \dot{x}(0)=2$$

试用 MATLAB 函数绘制系统的相平面图。

解：取状态变量 $x_2=x$，$x_1=\dot{x}$，得到系统状态方程模型，即一阶常微分方程组：

$$\begin{cases}\dot{x}_2=\dfrac{\mathrm{d}x}{\mathrm{d}t}=x_1\\ \dot{x}_1=\ddot{x}=x_1(1-x_2^{\,2})-x_2\end{cases}\quad 且\begin{cases}x_2(0)=2\\ x_1(0)=3\end{cases}$$

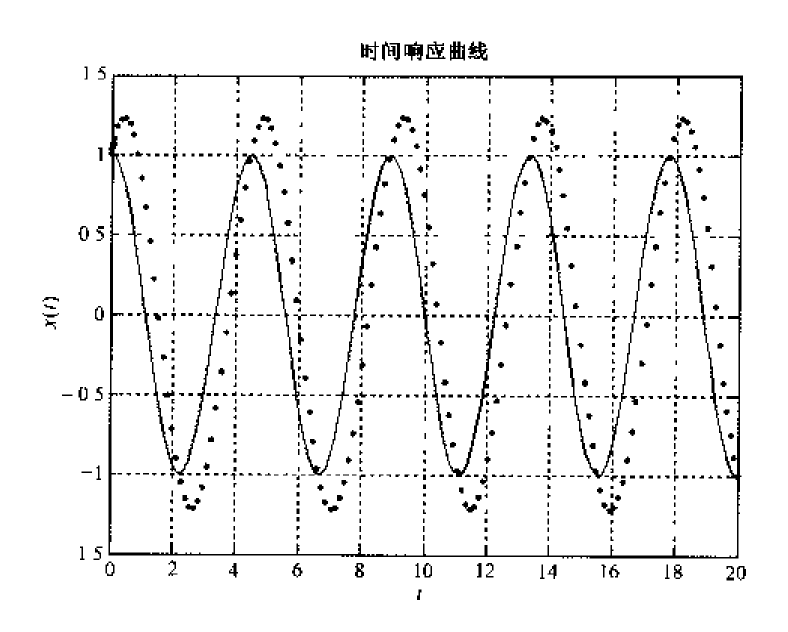


图 11.6　例 11-1 时域响应图

由此模型就可以用 MATLAB 来求解了。

主程序 MATLAB 代码如下：

```
% test_11_2 是系统微分方程的描述函数
[t, x]=ode45(' test_11_2', [0, 20], [3, 2])
plot(x(:, 1), x(:, 2))
axis([-6, 6, -6, 6])
xlabel('x')
ylabel('dx/dt')
grid
title('相轨迹图')
```

子函数 MATLAB 代码如下：

```
%状态导数
function xdot=  test_11_2(t, x)

xdot=[x(1)*(1-x(2)^2)-x(2); x(1)];
%   function xdot=filename(t, x)
%   xdot=[表达式 1；表达式 2；表达式 3；…；表达式 n-1]
%   表达式 1  对应   x1'=x2
%   表达式 2  对应   x2'=x3
%   表达式 3  对应   x3'=x4
%   ...
%   表达式 n-1  对应   xn-1'=xn
```

主函数运行后，输出如图 11.7 所示。

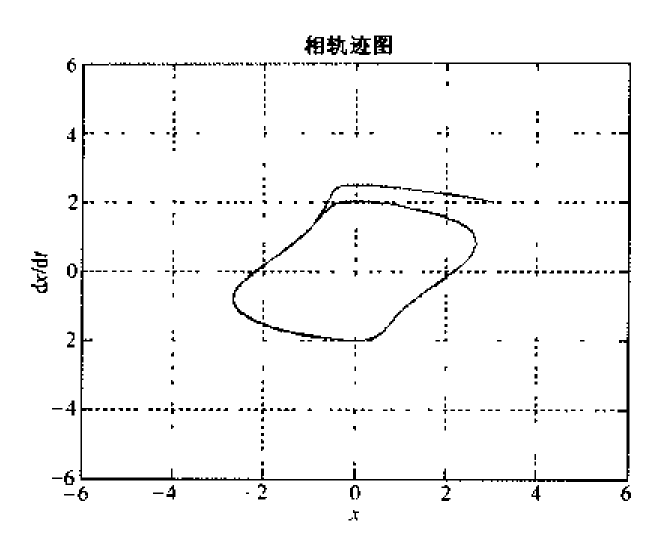


图 11.7 例 11-2 相轨迹图

11.3.2.2 采用 Simulink 绘制相轨迹图

Simulink 对于非线性系统的分析与设计是很有用的，Simulink 提供了非线性模块，包括死区、饱和、继电等多种类型的非线性，也能构成很复杂的非线性函数。

下面以实例说明如何在 Simulink 中使用非线性模块以及如何采用 Simulink 绘制相轨迹。

【例 11-3】 已知一个非线性控制系统如图 11.8 所示，输入为零初始条件，线性环节为 $G(s)=\dfrac{K}{s(Ts+1)}$，其中，$T=1$，$K=4$，N 为如图 11.2 所示的理想饱和非线性，

$$y=\begin{cases}-0.2 & x<-0.2\\ x & |x|\leqslant 0.2\\ 0.2 & x>0.2\end{cases}$$

，系统的初始状态为 0，试：

（1）在 $e-\dot{e}$ 平面上画出相轨迹；

（2）绘出 $e(t)$，$c(t)$ 的时间响应波形。

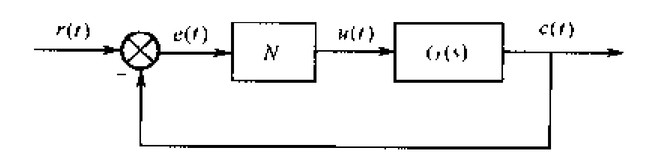


图 11.8 非线性控制系统示意图

解：取状态变量 $e(t)$ 和 $\dot{e}(t)$，使用 Simulink 来解此题的步骤如下。

在 MATLAB 7.0 的 MATLAB 窗中中双击 Simulink 图标就打开 Simulink Library Browser 窗口，再在此窗口进入 File\New\Model，打开一个 untitled 窗（可以用 Save as 保

存此窗口并改名）。

在 Simulink Library Browser 窗口下有 CONTINUOUS、DISCONTINUTIES、MATH OPERATIONS、SINK、SOURCE 等子目录，每个子目录下都包含若干可利用的模块，可直接拖至 untitled 窗口，如图 11.9 所示。

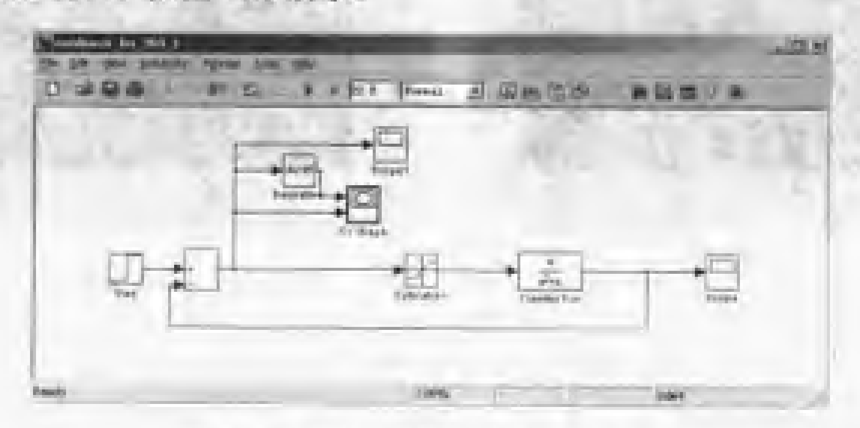


图 11.9　例 11-3 的 Simulink 仿真模型

在图 11.9 中，传递函数环节（Transfer Fcn）、微分环节（Deritivative）来自 Simulink\CONTINUOUS；饱和非线性（Saturation）来自 Simulink\DISCONTINUTIES；求和（Sum）来自 Simulink\MATH OPERATIONS；双踪示波器（XY Graph）、单踪示波器（Scope）来自 Simulink\SINK；阶跃函数（Step）来自 Simulink\SOURCE。

要在 XY Graph 上绘出相轨迹，关键是要得到 e 和 $\dot{e}$ 信号。显然，e 直接取自比较器的输出（即图中 Sum 环节的输出），$\dot{e}$ 可以在 e 后面加一微分环节实现，然后把这两个信号接到 XY Graph 便可画出相轨迹。

双击饱和模块，就会出现该模块的设置窗口，按照题意设置饱和特性的限幅为[-0.2，0.2]。

在 SIMULATION\SIMULATION PARAMETERS\SOLVER 中设置 SolverType 为“Fixed Step”，“Solver”（步长）为 0.05，“Stop Time”为 40。运行 SIMULATION\START，XY Graph 绘出的相轨迹如图 11.10 所示。

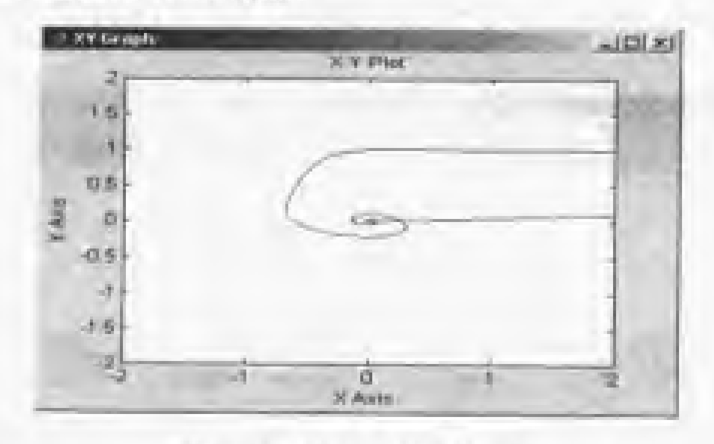


图 11.10　例 11-3 的相轨迹图

双击系统输出连接的单踪示波器，看到 $c(t)$ 的时间响应波形，如图 11.11 所示。

双击比较环节输出连接的单踪示波器，看到 $e(t)$ 的时间响应波形，如图 11.12 所示。



图 11.11　例 11-3 中 $c(t)$的时间响应波形



图 11.12　例 11-3$e(t)$的时间响应波形

【例 11-4】　已知一个控制系统如图 11.13 所示，输入为零初始条件，线性环节为 $G_2(s)=\dfrac{1}{s(4s+1)}$，系统的初始状态为 0。$G_1(s)$ 取下列两种情况：

（1）$G_1(s)$ 为非线性环节 N，且 N 为理想饱和非线性，它的函数为 $y=\begin{cases}-2, & x<-2\\ x, & |x|\leqslant 2\\ 2, & x>2\end{cases}$。

（2）$G_1(s)$ 为比例环节，比例增益为 2。

试求系统在单位阶跃作用下的相轨迹以及系统输出。

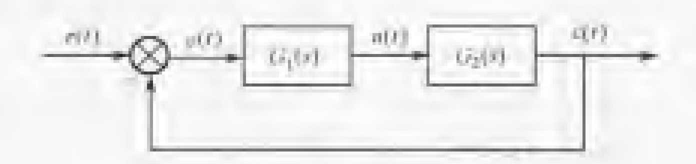


图 11.13　例 11-4 的控制系统框图

解：取状态变量 $e(t)$ 和 $\dot{e}(t)$，使用 Simulink 建立如图 11.14 所示的仿真框图。

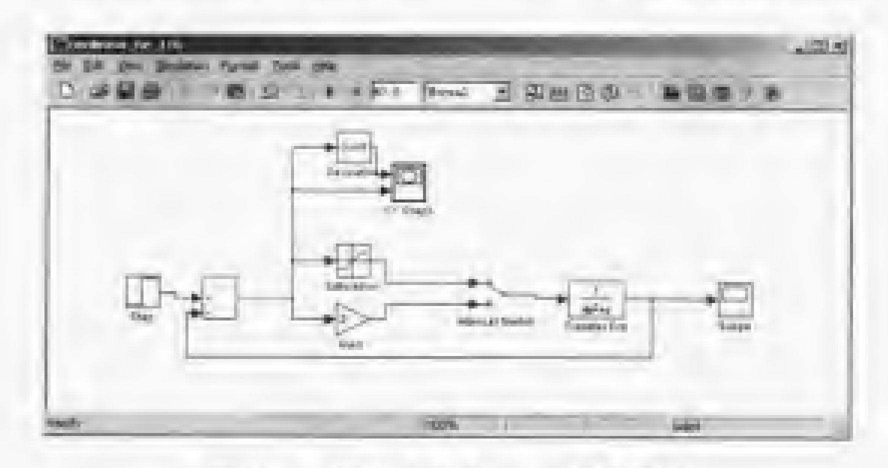


图 11.14　例 11-4 的 Simulink 仿真模型

在图 11.14 中，传递函数环节（Transfer Fcn）、微分环节（Deritivative）来自 Simulink\CONTINUOUS；饱和非线性（Saturation）来自 Simulink\DISCONTINUTIES；求和（Sum）来自 Simulink\MATH OPERATIONS；双踪示波器（XY Graph）、单踪示波器（Scope）来自 Simulink\SINK；阶跃函数（Step）来自 Simulink\SOURCE；手动切换开关（Manual Switch）来自 Simulink\SIGNAL ROUTING，它用来在同一个图上实现 $G_1(s)$ 两种情况的切换，双击该图标便可实现切换。

要在 XY Graph 上绘出相轨迹，关键是得到 e 和 $\dot{e}$ 信号，显然，e 直接取自比较器的输出（即图中的 Sum 环节的输出），$\dot{e}$ 可以在 e 后面加一个微分环节实现，然后把这两个信号接到 XY Graph 便可画出相轨迹。

双击饱和模块，就会出现该模块的设置窗口，按照题意设置饱和特性的限幅为[−2，2]。

在手动开关选择非线性模块后，运行 SIMULATION\START，XY Graph 绘出的相轨迹如图 11.15 所示。

双击系统输出连接的单踪示波器，看到系统的时间响应波形，如图 11.16 所示。

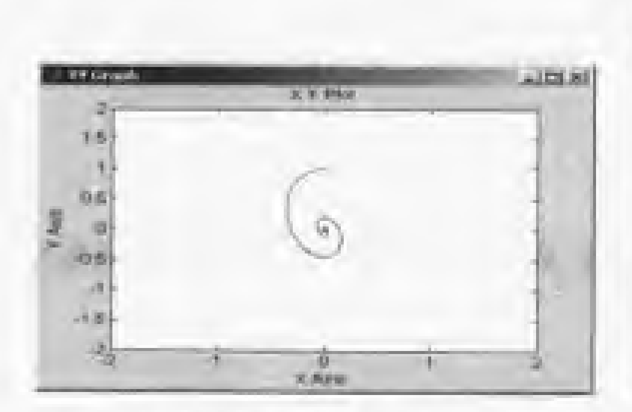


图 11.15　$G_1(s)$为非线性环节时的相轨迹图



图 11.16　$G_1(s)$为非线性环节时的系统输出

在手动开关选择线性模块后，运行 SIMULATION\START，XY Graph 绘出的相轨迹如图 11.17 所示。

双击系统输出连接的单踪示波器，看到系统的时间响应波形，如图 11.18 所示。

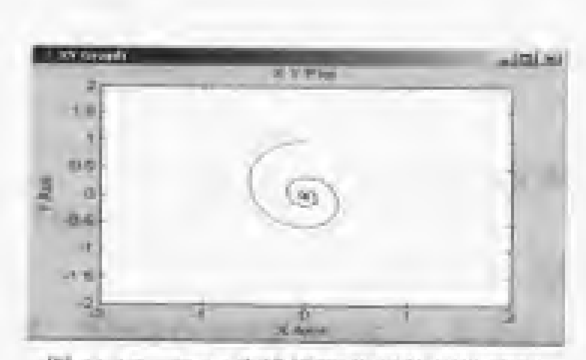


图 11.17　$G_1(s)$为线性环节时的相轨迹图



图 11.18　$G_1(s)$为线性环节时的系统输出

11.4　描述函数法

描述函数法是达尼尔（P.J.Daniel）于 1940 年提出的，它是线性系统频率法在非线性系统中的推广，是非线性系统稳定性的近似判别法，它要求系统具有良好的低通特性且非线性较弱，描述函数法的优点是能用于高阶系统。

11.4.1　描述函数基本概念

在频率法中，对于线性时不变系统，当输入为正弦函数时输出也是同频率的正弦函数，输出和输入只有幅值和相位的差别。对于非线性系统，当输入为正弦函数时，输出是同频率的非正弦函数，也就是说输出中含有高次谐波，可见线性系统的频率法不适用于非线性系统。描述函数的作用相当于对线性系统中的频率法进行改进并使之适用于非线性系统。

描述函数的基本思想：当系统满足一定的假设条件时，系统中非线性环节在正弦信号作用下的输出可用一次谐波分量来近似，由此导出非线性环节的近似等效频率特性，即描述函数。这样，非线性系统就近似等效为一个线性系统，并可用线性系统理论中的频率法对系统进行频域分析。

11.4.2　描述函数定义

对于如图 11.19 所示的非线性系域，其中 N 是非线性环节，它的输入/输出关系为 $y(t)=f(x(t))$，线性部分的传递函数为 $G(s)$。

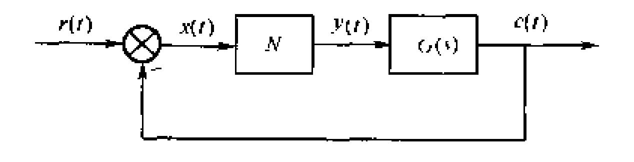


图 11.19　非线性系统框图

当非线性环节的输入为如下正弦函数时：

$$x(t)=X\sin\omega t \tag{11-10}$$

其中，X 是正弦函数的幅值。

将非线性环节的输出 $y(t)$ 分解为傅里叶级数：

$$\begin{aligned} y(t) &= A_0+\sum_{n=1}^{\infty}(A_n\cos n\omega t+B_n\sin n\omega t) \\ &= A_0+\sum_{n=1}^{\infty}Y_n\sin(n\omega t+\phi_n) \end{aligned} \tag{11-11}$$

其中，A_0 是直流分量；A_n, B_n 是傅里叶系数；Y_n, φ_n 分别是第 n 次谐波分量的幅值和相角，且

$$
\begin{aligned}
A_n &= \frac{1}{\pi}\int_0^{2\pi} y(t)\cos n\omega t \mathrm{d}(\omega t) \\
B_n &= \frac{1}{\pi}\int_0^{2\pi} y(t)\sin n\omega t \mathrm{d}(\omega t) \\
Y_n &= \sqrt{A_n^2 + B_n^2} \\
\phi_n &= \tan^{-1}(\frac{A_n}{B_n})
\end{aligned} \tag{11-12}
$$

如果非线性特性是奇对称的，那么直流分量 $A_0 = 0$，式（11-11）可简化为：

$$y(t) = \sum_{n=1}^{\infty}(A_n \cos n\omega t + B_n \sin n\omega t) = \sum_{n=1}^{\infty} Y_n \sin(n\omega t + \phi_n) \tag{11-13}$$

式（11-13）表明，非线性环节的输出 $y(t)$ 中含有高次谐波。如果系统线性部分具有良好的低通滤波特性，那么系统信号中高次谐波就被大大衰减，即当 $n>1$ 时 Y_n 会很小，因此可以用基波来近似，如式（11-14）所示：

$$y(t) \approx y_1(t) = A_1 \cos\omega t + B_1 \sin\omega t = Y_1 \sin(\omega t + \phi_1) \tag{11-14}$$

可以看出，非线性环节可近似为与线性环节类似的频率响应形式，这就是非线性特性在频域的线性化。

在正弦信号输入下可用非线性环节输出中的基波分量和输入正弦函数的复数比来描述这个非线性环节，如式（11-15）所示：

$$N(X) = \frac{B_1(X) + \mathrm{j}A_1(X)}{X} = \frac{Y_1(X)}{X}\mathrm{e}^{\mathrm{j}\phi_1(X)} \tag{11-15}$$

其中，幅值 X 是一个待定常数；A_1, B_1, Y_1, ϕ_1 只与 X 有关，记为 $A_1(X), B_1(X), Y_1(X)$ 和 $\phi_1(X)$；$N(X)$ 称为非线性环节 $y(t) = f(x(t))$ 的描述函数。

显然，描述函数是幅值 X 的函数，描述函数可以理解为非线性环节在忽略高次谐波情况下的非线性增益——这个增益与输入正弦函数的幅值有关。如果非线性特性是单值奇对称的，那么 $A_1 = 0, \phi_1 = 0, N = B_1 / X$。

这相当于用一个等效环节代替原来的非线性环节，而等效环节的幅相特性函数 $N(X)$ 是输入函数 $x(t) = X\sin\omega t$ 幅值 X 的函数，这样，图 11.19 就可等效为图 11.20。

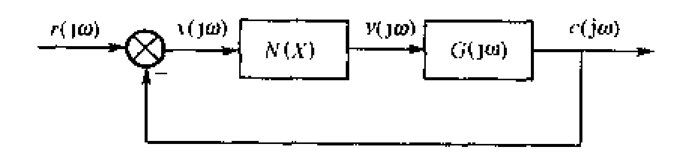


图 11.20　非线性系统等效框图

11.4.3　常见非线性特性描述函数

当非线性环节的频率特性用描述函数表示后，线性系统中普遍应用的频率法就可以推广到非线性系统了，问题的关键是非线性环节描述函数如何计算。

描述函数求解的基本过程是：先根据已知的输入 $x(t) = X\sin\omega t$ 和非线性特性

$y(t)=f(x(t))$ 求出输出 $y(t)$，然后通过积分求出 $A_1(X),B_1(X),Y_1(X),\phi_1(X)$ 以及 $N(X)$。其工作量和技巧主要在积分。

下面举例说明描述函数的计算过程。

对于如图 11.1 所示的死区非线性，当输入为正弦函数 $x(t)=X\sin\omega t(X>\Delta)$ 时，输出 $y(t)$为：

$$y(t)=\begin{cases}0 & 0\leqslant \omega t\leqslant \omega t_1\\ k(X\sin\omega t-\Delta) & \omega t_1\leqslant \omega t\leqslant \dfrac{\pi}{2}\end{cases}$$

其中 $\omega t_1=tg^{-1}\Delta$，由于图中的死区非线性是单值奇对称的，因此 $A_1=0,\phi_1=0$，且

$$B_1=\frac{1}{\pi}\int_0^{2\pi}y(t)\sin\omega t\mathrm{d}(\omega t)=\frac{4}{\pi}\int_0^{\frac{\pi}{2}}y(t)\sin\omega t\mathrm{d}(\omega t)$$
$$=\frac{4k}{\pi}\int_0^{\frac{\pi}{2}}(X\sin\omega t-\Delta)\sin\omega t\mathrm{d}(\omega t)$$

其中，$\Delta=X\sin\omega t_1$，$\omega t_1=\sin^{-1}\left(\dfrac{\Delta}{X}\right)$；

故

$$B_1=\frac{4Xk}{\pi}\left[\int_{\omega t_1}^{\frac{\pi}{2}}\sin^2\omega t-\sin\omega t_1\int_{\omega t_1}^{\frac{\pi}{2}}\sin\omega t\mathrm{d}(\omega t)\right]$$
$$=\frac{4Xk}{\pi}\left[\int_{\omega t_1}^{\frac{\pi}{2}}\frac{1-\cos 2\omega t}{2}-\sin\omega t_1\int_{\omega t_1}^{\frac{\pi}{2}}\sin\omega t\mathrm{d}(\omega t)\right]$$
$$=\frac{4Xk}{\pi}\left[\left(\frac{\pi}{4}-\frac{\omega t_1}{2}+\frac{1}{2}\sin\omega t_1\cos\omega t_1\right)-\sin\omega t_1\cos(\omega t_1)\right]$$
$$=\frac{2Xk}{\pi}\left[\frac{\pi}{2}-\omega t_1-\sin\omega t_1\cos\omega t_1\right]$$
$$=\frac{2Xk}{\pi}\left[\frac{\pi}{2}-\sin^{-1}(\frac{\Delta}{X})-\frac{\Delta}{X}\sqrt{1-\left(\frac{\Delta}{X}\right)^2}\right]$$

代入即可得理想死区非线性的描述函数，如下所示：

$$N(X)=\frac{B_1}{X}=k-\frac{2k}{\pi}\left[\sin^{-1}\left(\frac{\Delta}{X}\right)-\frac{\Delta}{X}\sqrt{1-\left(\frac{\Delta}{X}\right)^2}\right]$$

表 11.1 列出了一些典型非线性描述函数，以供查用。

表 11.1 典型非线性描述函数

非线性类型	静 特 性	描述函数 $N(X)$
理想继电非线性	$y=\begin{cases}+M, & x>0\\ -M, & x<0\end{cases}$	$\dfrac{4M}{\pi X}$
饱和非线性	$y=\begin{cases}-kg & x<-g\\ kx & \lvert x\rvert\leqslant g\\ kg & x>g\end{cases}$	$\dfrac{2k}{\pi}\left[\arcsin\dfrac{g}{X}+\dfrac{g}{X}\sqrt{1-\left(\dfrac{g}{X}\right)^2}\right]$，$X\geqslant g$

续表

非线性类型	静　特　性	描述函数 $N(X)$
间隙非线性	$y=\begin{cases} k(x-b) & x-y/k=b \\ 0 & -b<x-y/k<b \\ k(x+b) & x-y/k=-b \end{cases}$	$\frac{k}{\pi}\left[\arcsin\left(1-\frac{2b}{X}\right)+2\left(1-\frac{2b}{X}\right)\sqrt{\frac{b}{X}\left(1-\frac{b}{X}\right)}\right.$ $\left.+\mathrm{j}\frac{4kb}{\pi X}\left(\frac{b}{X}-1\right)+\frac{\pi}{2}\right], X \geqslant b$
有死区与滞环的继电非线性	$x>0,\ y=\begin{cases} +M, & x>h \\ 0, & h>x>-mh \\ -M, & x<-mh \end{cases}$ $x<0,\ y=\begin{cases} +M, & x>mh \\ 0, & mh>x>-b \\ -M, & x<-h \end{cases}$	$\frac{2M}{\pi X}\left[\sqrt{1-\left(\frac{mh}{X}\right)^2}+\sqrt{1-\left(\frac{h}{X}\right)^2}\right]$ $+\mathrm{j}\frac{2Mh}{\pi X^2}(m-1),\ X \geqslant h$

11.4.4 稳定性分析

对于如图 11.21 所示的非线性控制系统，N 表示非线性部分，$G(s)$表示线性部分，

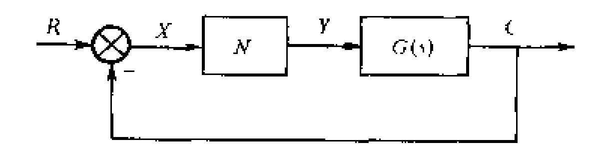


图 11.21　非线性系统框图

则闭环系统频率特性为：

$$\frac{C(\mathrm{j}\omega)}{R(\mathrm{j}\omega)}=\frac{N(\mathrm{j}\omega)G(\mathrm{j}\omega)}{1+N(\mathrm{j}\omega)G(\mathrm{j}\omega)} \tag{11-16}$$

特征方程为：

$$1+N(\mathrm{j}\omega)G(\mathrm{j}\omega)=0 \tag{11-17}$$

如果非线性部分用放大系数 k 来表示，即 $N(X)=k$，则特征方程与线性系统的特征方程相似。

式（11-17）可表示为：

$$-N(\mathrm{j}\omega)=\frac{1}{G(\mathrm{j}\omega)} \tag{11-18}$$

或

$$G(\mathrm{j}\omega)=-\frac{1}{N(\mathrm{j}\omega)} \tag{11-19}$$

式（11-18）用线性部分的频率特性的逆与负描述函数特性比较，式（11-19）用负倒描述函数与频率特性比较，它们常用于非线性系统的稳定性分析。

非线性系统稳定条件为：线性部分频率特性 $G(\mathrm{j}\omega)$ 的逆应包围非线性部分描述函数负值 $-N(\mathrm{j}\omega)$ 的点；或者，线性部分的（$-1, 0$）稳定点的判别被描述函数的负倒数代替。将描述函数的负倒数的轨迹作为判别点轨迹，则系统稳定判据如下：

（1）在复平面上，如果曲线 $G(\mathrm{j}\omega)$ 不包围 $-\dfrac{1}{N(\mathrm{j}\omega)}$ 曲线，那么闭环系统稳定，两者的距离越远，系统越稳定；

（2）在复平面上，如果 $G(\mathrm{j}\omega)$ 曲线包围 $-\dfrac{1}{N(\mathrm{j}\omega)}$ 曲线，那么闭环系统不稳定；

（3）在复平面上，如果曲线 $G(\mathrm{j}\omega)$ 与曲线 $-\dfrac{1}{N(\mathrm{j}\omega)}$ 相交，那么闭环系统临界稳定，闭环系统出现自振荡（极限环）。

为方便起见，通常将曲线 $-\dfrac{1}{N(\mathrm{j}\omega)}$ 称为“负倒描述函数曲线”。

11.5　采用 Simulink 分析非线性系统

【例 11-5】　设非线性控制系统如图 11.22 所示。

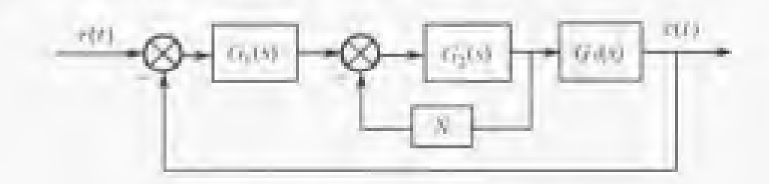


图 11.22　例 11-5 的系统框图

其中 $G_1(s)=5$，$G_2(s)=\dfrac{1}{0.5s+1}$，$G_3(s)=\dfrac{1}{s}$，非线性环节 N 为死区非线性，其表达式为：

$$y=\begin{cases}x+2 & x<-2\\ 0 & |x|\leqslant 2\\ x-2 & x>2\end{cases}$$

试用 Simulink 分析系统单位阶跃响应，并绘制响应曲线。

解：使用 Simulink 建立仿真框图，如图 11.23 所示。

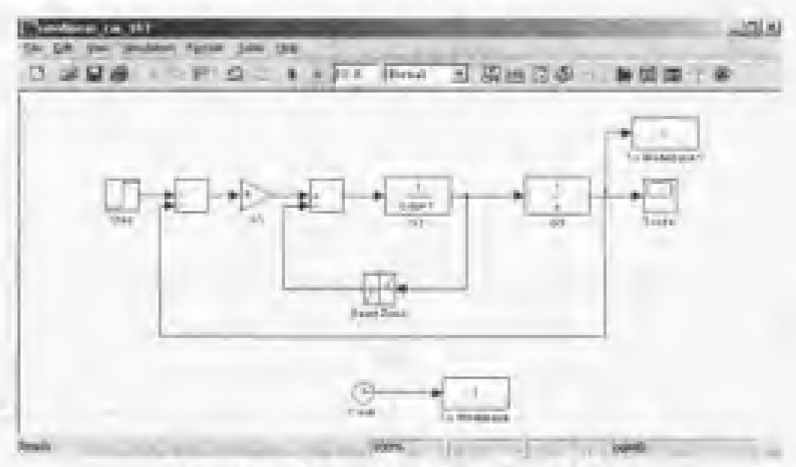


图 11.23　例 11-5 的 Simulink 仿真模型

在图 11.23 中，传递函数环节（Transfer Fcn）、微分环节（Deritivative）来自 Simulink\CONTINUOUS；死区非线性（Dead Zone）来自 Simulink\DISCONTINUTIES；求和（Sum）来自 Simulink\MATH OPERATIONS；双踪示波器（XY Graph）、单踪示波器（Scope）来自 Simulink\SINK；阶跃函数（Step）来自 Simulink\SOURCE；输出到工作空间（To Workspace）来自 Simulink\SINK，用鼠标双击“To Workspace”图标，得到如图 11.24 所示的 To Workspace 模块参数对话框。

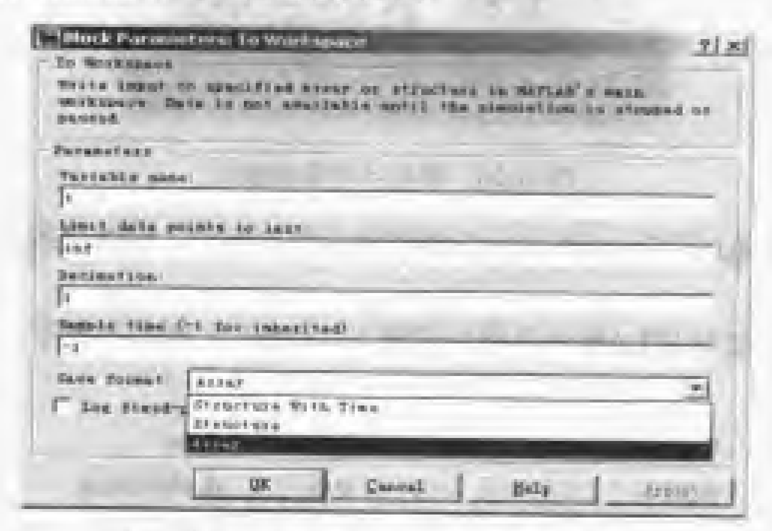


图 11.24　To Workspace 模块参数对话框

本题中，需要传输数据向量为 $c(t)$和 t，以设置数据向量 t 为例，在 Variable name 编辑框中输入向量名“t”，save format 编辑选择“Array”（向量）项，然后点击“OK”按钮完成设置。仿真运行后，向量 $c(t)$和 t 以各自变量名存在于 MATLAB 工作空间中，以供 MATLAB 程序使用。

双击死区模块，就会出现该模块的设置窗口，如图 11.25 所示，图中显示的死区范围是默认值±0.5，分别在 Start of dead zone 和 End of dead zone 编辑框内输入“−2”和“+2”，其他选项按默认值设定，然后单击“OK”按钮完成设置。

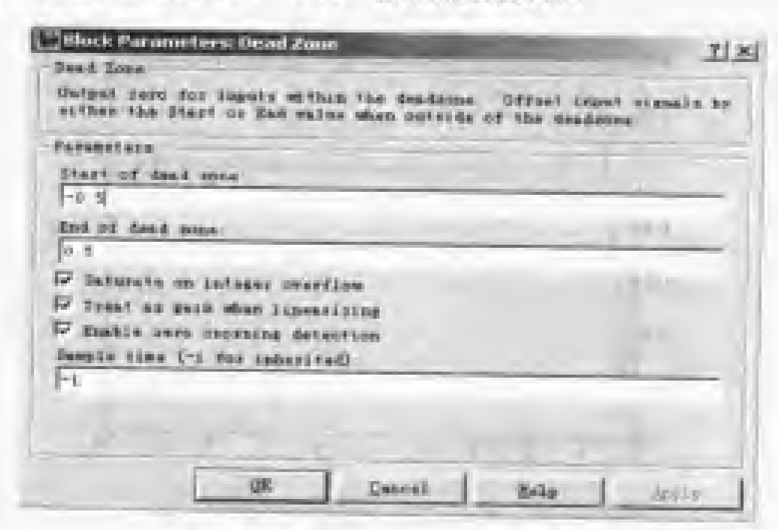


图 11.25　Dead zone 模块参数对话框

启动仿真，双击示波器，得到如图 11.26 所示的图形，它就是系统的阶跃响应曲线。

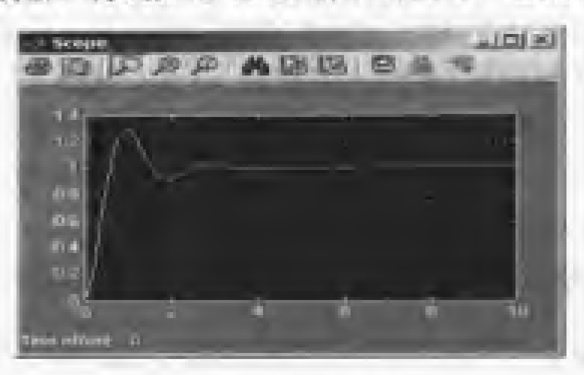


图 11.26　例 11-5 的系统输出

由于输出结果通过“To workspace”传送到工作空间中，因此系统输出响应曲线也可在工作空间中显示。

启动仿真后，在 MATLAB 命令窗口中输入“whos”，则显示如下：

```
>> whos
Name      Size                 Bytes    Class
  c        62x1                 496      double array
  t        62x1                 496      double array
  tout     62x1                 496      double array
Grand total is 186 elements using 1488 bytes
```

然后输入画图命令，即可将系统输出响应曲线绘制出来。

在 MATLAB 命令窗口中输入如下程序：

```
plot(t, c)
xlabel('t')
ylabel('c(t)')
```

回车后显示如图 11.27 所示的响应曲线。

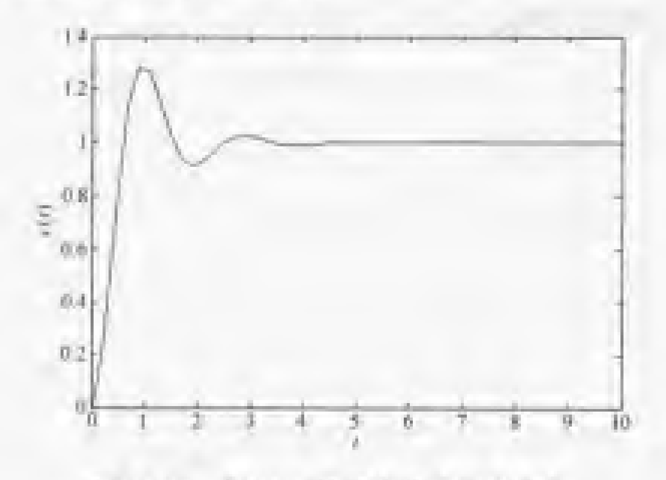


图 11.27　例 11-5 的系统阶跃响应曲线

第 12 章　离散控制系统

12.1　引　　言

近年来，随着脉冲技术、数字式元器件、数字电子计算机，特别是微处理机的迅速发展，数字式、离散式控制器在许多场合取代了模拟式、连续式控制器。数字、离散控制器的应用使控制系统发生了根本变化。随着数字计算机的不断发展，离散控制理论与技术不断深化，其应用范围越来越广。

本章主要描述离散控制基本概念及理论知识，介绍计算机控制的基础理论、离散控制系统分析和设计的基本方法，以及 MATLAB/Simulink 在离散控制系统中的应用。通过本章，读者对数字、离散控制系统能有一个比较全面的认识。

12.2　离散控制系统基本概念

12.2.1　离散控制系统概述

数字、离散控制系统与连续控制系统的根本区别在于：

（1）离散控制系统中既可以包含连续信号，又可以包含离散信号，是一个混合信号系统。

（2）连续系统中的控制信号、反馈信号以及偏差信号都是连续型的时间函数，而在离散系统中则不然。一般情况下，其控制信号是离散型的时间函数，因此取自系统输出端的负反馈信号在和离散控制信号进行比较时，同样需要采用离散型的时间函数，那么比较后得到的偏差信号也将是离散型的时间函数。

（3）分析和设计数字、离散控制系统的数学工具是 Z 变换，采用的数学模型是差分方程、脉冲传递函数。

12.2.1.1　离散控制系越基本组成

离散控制系统结构形式多样，一般如图 12.1 所示，主要由采样器、数字控制器、保持器、执行器、被控对象和测量变送器构成。

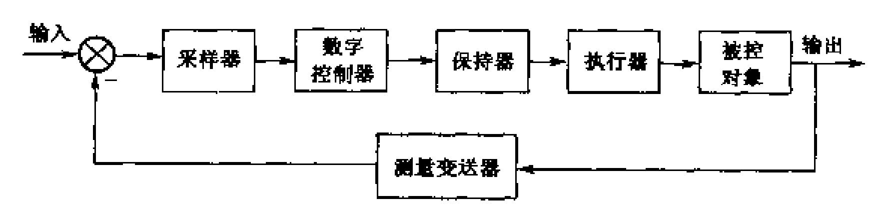


图 12.1　离散控制系统结构示意图

（1）采样器：将连续信号转换成脉冲序列。

（2）数字控制器：常用的是数字计算机，构成控制系统的数字部分，对系统进行控制，通过这部分的信号均以离散形式出现。

（3）保持器：将数字控制器输出的离散信号转换成模拟信号，用来实现采样点之间的插值，常见的保持器有零阶保持器和一阶保持器。

（4）执行器：根据控制器的控制信号，改变输出的角位移或直线位移，并通过调节机构改变被调介质的流量或能量，使工作过程符合预定要求。按照不同的动力方式可分为电动执行器、气动执行器和液动执行器。

（5）被控对象：所要控制的装置或设备。

（6）测量变送器：通常由传感器和测量线路构成，用以将被控参数转换成某种形式的信号。

12.2.1.2　数字控制系统工作过程

数字控制系统通过数字计算机闭合而成，包括工作于离散状态下的数字计算机（或专用数字控制器）和具有连续工作状态的被控对象两大部分，其工作过程如图 12.2 所示。图中虚线内为用于控制目的的数字计算机或数字控制器，它构成控制系统的数字部分，通过这部分的信号均以离散形式出现；被控对象一般用 $G(s)$表示，是系统的不可变部分，它是控制系统连续部分的主要成分。

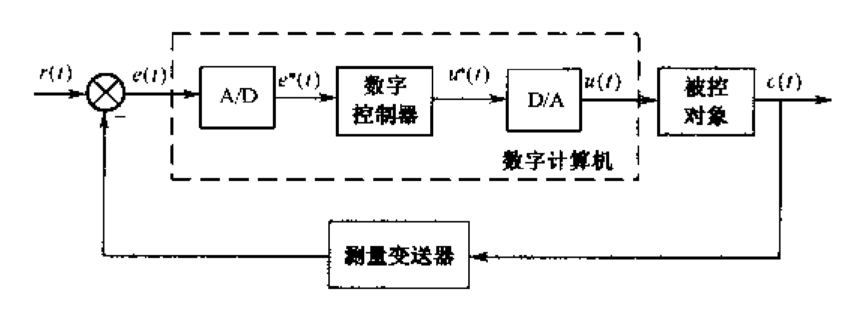


图 12.2　数字控制系统工作示意图

在数字控制系统中，具有连续时间函数形式的被控信号 $c(t)$（模拟量）受控于具有离散时间函数形式的控制信号 $u^*(t)$（数字量）。为了实现控制，需要通过数模转换环节（D/A）将数字量转换为模拟量，即将 $u^*(t)$ 转换为 $u(t)$ 。连续的被控信号 $c(t)$ 经反馈环节反馈至输入端并与参考输入 $r(t)$ 进行比较，得到 $e(t)$ ， $e(t)$ 经模数转换环节（A/D）得到偏差信号 $e^*(t)$（数字量）。离散的偏差信号 $e^*(t)$ 经数字计算机加工处理变换成数字信号 $u^*(t)$ ， $u^*(t)$ 再经 D/A 转换为连续信号 $u(t)$ ， $u(t)$ 馈送到连续部分的执行机构去控制系统的被控信号 $c(t)$ 。

12.2.1.3　离散控制系统基本特点

离散控制系统在自动控制领域中越来越多地被广泛应用，它具有以下基本特点：

（1）以数字计算机为核心组成数字式控制器，可实现复杂的控制要求，控制效果好，并可通过软件方式改变控制规律，控制方式灵活；

（2）数字信号传输可有效抑制噪声，提高系统的抗干扰能力；

（3）可采用高灵敏度的控制元件，提高系统的控制精度；

（4）可用一台数字计算机实现对几个系统的分时控制，提高设备利用率，经济性好；

（5）对于大滞后、大惯性系统，具有较好的控制效果；

（6）便于组成功能强大的集散控制系统。

12.2.1.4 离散控制系统研究方法

数字、离散控制系统的研究主要方法有时域分析法和频域分析法。

（1）时域分析法：通过建立时间域中离散输入序列与离散输出序列之间逻辑关系的数学模型，建立 n 阶线性差分方程式并求解。

（2）频域分析法：以 Z 变换理论为基础，通过建立反映离散系统输入/输出特性的脉冲传递函数，将连续控制系统的分析计算方法用于离散控制系统的分析计算。

线性离散控制系统和线性连续控制系统的研究方法类似，因此，可进行一些类比和分析以便借鉴使用，如表 12.1、表 12.2、表 12.3 所示。

表 12.1 线性连续控制系统与线性离散控制系统研究方法比较 1

系统类型 / 比较内容	线性连续控制系统	线性离散控制系统
数学描述	线性微分方程	线性差分方程
变换方法	拉普拉斯变换	离散拉普拉斯变换或 Z 变换
瞬态响应	与闭环极点和零点在 S 平面分布有关	与闭环极点和零点在 Z 平面分布有关
稳定充要条件	闭环极点全部位于 S 平面的左半部	闭环极点全部位于 Z 平面以原点为圆心的单位圆内
传递函数	$G_c(s)=\dfrac{Y(s)}{R(s)}$	$G_c(z)=\dfrac{Y(z)}{R(z)}$
过渡函数	设脉冲响应函数为 $h(t)$，输入函数为 $r(t)$，则输出函数为 $y(t)=h(t)\cdot r(t)$	设脉冲响应函数为 $h(kT)$，输入函数为 $h(kT)$，则输出函数为 $y(kT)=h(kT)\cdot r(kT)$

表 12.2 线性连续控制系统与线性离散控制系统研究方法比较 2

系统类型 / 比较内容		线性连续控制系统	线性离散控制系统
频率法	频率特性	$G_O(s)\big\|_{s=j\omega}\to G_O(j\omega)$	$G_O(z)\big\|_{z=e^{j\omega T}}\to G_O(e^{j\omega T})$
	对数频率特性	$20\lg\|G_O(j\omega)\|\sim\lg\omega$ $\varphi(\omega)\sim\lg\omega$	$G_O(z)\big\|_{z=\frac{1+jv}{1-jv}}\to G_O(jv)$ $20\lg\|G_O(jv)\|\sim\lg v$，$\varphi(v)\sim\lg v$
根轨迹法	幅值条件	$\|G_O(s)\|=1$	$\|G_O(z)\|=1$
	相角条件	$\angle G_O(s)=\pm180^\circ+i\cdot360^\circ$ $(i=0,1,2,3,\ldots)$	$\angle G_O(z)=\pm180^\circ+i\cdot360^\circ$ $(i=0,1,2,3,\ \ )$
	绘制法则	在 S 平面上绘制	在 Z 平面上绘制，绘制法则与线性连续控制系统类似

表 12.3　线性连续控制系统与线性离散控制系统研究方法比较 3

比较内容 \ 系统类型		线性连续控制系统	线性离散控制系统
状态空间法	状态空间表达式	$x(t)=Ax(t)+Bu(t)$ $y(t)=Cx(t)+Du(t)$	$x(kT+T)=Fx(kT)+Gu(kT)$ $y(kT)=Cx(kT)+Du(kT)$
	传递矩阵	$G(s)=H(s)=C[sI-A]^{-1}B+D$	$G(z)=H(z)=C[zI-F]^{-1}G+D$
	特征方程	$\lvert sI-A\rvert=0$	$\lvert zI-A\rvert=0$
	状态方程求解	$x(t)=e^{At}x(0)+\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau$ $x(t)=L^{-1}[(sI-A)^{-1}]x(0)+L^{-1}[(sI-A)^{-1}BU(s)]$	$x(kT)=F^k x(0)+\sum_{j=0}^{k-1}F^{k-j-1}Gu(jT)$ $x(kT)=Z^{-1}[(zI-F)^{-1}zx(0)]+Z^{-1}[(zI-F)^{-1}GU(z)]$
	稳定充要条件	在 S 平面上绘制	在 Z 平面上绘制，绘制法则与线性连续控制系统类似

12.2.2　离散信号的数学描述

离散系统的一个显著特点是系统中一处或多处信号是脉冲序列或数字序列，而自然界的信号多是连续信号，为了把连续信号变为脉冲信号，需要对连续信号进行采样，为了把脉冲信号变为连续信号则需要用保持器。

12.2.2.1　采样过程及采样定理

1. 采样过程

采样过程是指采样器按一定的时间间隔对连续信号 $e(t)$ 进行采样，将其转换为相应的脉冲序列，即采样信号 $e^*(t)$ 的获取过程。实现采样过程的装置称为采样器或采样开关。

采样器可以用一个周期性闭合的开关来表示，其闭合周期为 T，每次闭合时间为 τ。实际上，由于采样持续时间通常远小于采样周期，即 $\tau \ll T$，也远小于系统连续部分的时间常数，因此，在分析采样系统时，可近似认为 $\tau \to 0$。在这种假设条件下，当采样开关的输入信号为连续信号 $e(t)$ 时，其输出信号 $e^*(t)$ 是一个脉冲序列，采样瞬时 $e^*(t)$ 的幅值等于相应瞬时 $e(t)$ 的幅值，即 $e(0), e(T), e(2T), \cdots, e(nT)$，采样过程如图 12.3 所示。

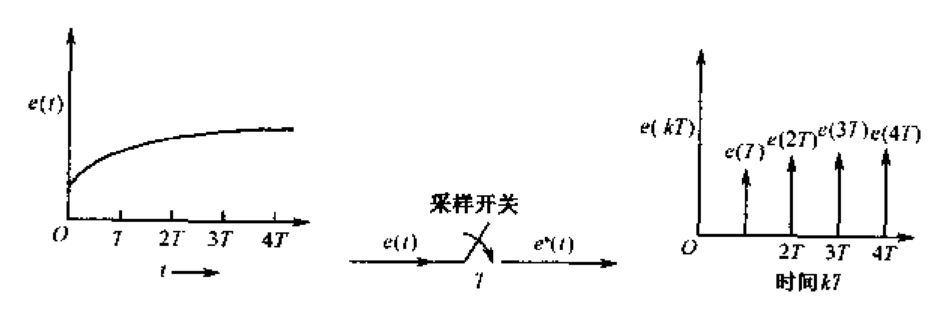


图 12.3　采样过程示意图

采样过程可以看成一个脉冲调制过程。理想的采样开关相当于一个单位理想脉冲序列发生器，它能够产生一系列单位脉冲 $e(0), e(T), e(2T), \cdots, e(nT)$ 。

$$
\begin{aligned}
e^*(t) &= e(0)[1(t)-1(t-\tau)] + e(T)[1(t-T)-1(t-T-\tau)] \\
&\quad + e(2T)[1(t-2T)-1(t-2T-\tau)] + \cdots \\
&= \sum_{k=0}^{\infty} e(kT)[1(t-kT)-1(t-kT-\tau)] \qquad (12\text{-}1) \\
&= \sum_{k=0}^{\infty} e(kT)\tau \frac{1(t-kT)-1(t-kT-\tau)}{\tau}
\end{aligned}
$$

当 $\tau \to 0$ 时，式（12-1）可写成：

$$e^*(t) = \sum_{k=0}^{\infty} e(kT)\delta(t-kT) \qquad (12\text{-}2)$$

或

$$e^*(t) = e(t)\sum_{k=0}^{\infty}\delta(t-kT) = e(t)\delta_T(t) \qquad (12\text{-}3)$$

2. 采样定理

采样定理（Shannon 定理）是在设计离散系统时必须要遵循的准则，它给出了自采样的离散信号不失真地恢复原连续信号所必需的理论上的最低采样频率。

对于如式（12-3）描述的采样信号 $e^*(t)$ ，令 $\delta_T(t) = \sum_{k=0}^{\infty}\delta(t-kT)$ ，则 $e^*(t)$ 可写成：

$$e^*(t) = e(t)\delta_T(t) = e(t)\frac{1}{T}\sum_{k=0}^{\infty} e^{jn\omega_s t}, \quad (t-nT<0, \delta(t-nT)=0) \qquad (12\text{-}4)$$

对 $e^*(t)$ 进行拉氏变换，可得：

$$E^*(s) = \frac{1}{T}\sum_{n=-\infty}^{\infty} E[s + jn\omega_s] \qquad (12\text{-}5)$$

式（12-5）表明，采样函数的拉氏变换式 $E^*(s)$ 是以 ω_s （ $\omega_s = \frac{2}{T}$ ，称为采样角频率）为周期的周期函数；它还表示了采样函数的拉氏变换式 $E^*(s)$ 与连续函数拉氏变换式 $E(s)$ 之间的关系。

通常 $E^*(s)$ 的全部极点均位于 S 平面的左半部，因此可用 $j\omega$ 代替上式中的复变量 s，直接求得采样信号的傅里叶变换：

$$E^*(j\omega) = \frac{1}{T}\sum_{n=-\infty}^{\infty} E[j(\omega + n\omega_s)] \qquad (12\text{-}6)$$

式（12-6）即为采样信号的频谱函数，它也反映了离散信号频谱和连续信号频谱之间的关系。

一般说来，连续函数的频谱是孤立的，其带宽是有限的，即上限频率为有限值，而离散函数 $e^*(t)$ 则具有以 ω_s 为周期的无限多个频谱。

在离散函数的频谱中， $n=0$ 的部分 $\frac{E(j\omega)}{T}$ 称为主频谱，它对应于连续信号的频谱。

除了主频谱外，$E^*(\mathrm{j}\omega)$ 还包含无限多个附加的高频频谱。为了准确复现所采样的连续信号，必须使采样后的离散信号的频谱彼此不重叠，这样就可以用一个比较理想的低通滤波器滤掉全部附加的高频频谱分量，保留主频谱。

相邻两频谱互不重迭的条件是：

$$\omega_s \geqslant 2\omega_{\max} \tag{12-7}$$

若满足式（12-7）的条件，并把采样后的离散信号 $e^*(t)$ 加到理想滤波器上，则在滤波器的输出端将不失真地复现原连续信号（幅值相差 $\frac{1}{T}$ 倍）。若 $\omega_s < 2\omega_{\max}$（$2\omega_{\max}$ 为连续信号的有限频率），则会出现相邻频谱的重叠现象，这时即使使用理想滤波器也不能将主频谱分离出来，因而难以准确复现原有的连续信号。

综上可以得到一条重要结论，即只有在 $\omega_s \geqslant 2\omega_{\max}$ 的条件下，采样后的离散信号 $e^*(t)$ 才有可能无失真地恢复为原来的连续信号，这就是香农（Shannon）采样定理。

12.2.2.2　保持器的数学描述

保持器是把数字信号转换成连续信号的装置，从数学上说，它解决了两个采样点之间的插值问题，即根据过去或现在的采样值进行外推，是一种时域的外推装置。

由采样过程的数学描述可知，在采样时刻上，连续信号的函数值与脉冲序列的脉冲强度相等，在 nT 时刻，有

$$e(t)\big|_{t=nT} = e(nT) = e^*(nT) \tag{12-8}$$

而在 $(n+1)T$ 时刻，则有

$$e(t)\big|_{t=(n+1)T} = e\left[(n+1)T\right] = e^*\left[(n+1)T\right] \tag{12-9}$$

然而，由脉冲序列 $e^*(t)$ 向连续信号 $e(t)$ 的转换过程中，在 nT 和 $(n+1)T$ 时刻之间，即当 $0<\Delta t<T$ 时，连续信号 $e(nT+\Delta t)$ 的值是多少？它与 $e(nT)$ 有何关系？这就是保持器要解决的问题。

通常把具有恒值、线性和抛物线外推规律的保持器分别称为零阶、一阶和二阶保持器。其中最简单、最常用的是零阶保持器和一阶保持器，下面分别进行介绍。

1. 零阶保持器

零阶保持器是一种按照恒值规律外推的保持器，它把前一采样时刻 nT 的采样值 $e(nT)$（在各采样点上 $e^*(nT)=e(nT)$）不增不减地保持到下一采样时刻 $(n+1)T$ 到来之前，从而使采样信号 $e^*(t)$ 变成阶律信号 $e_h(t)$，如图 12.4 所示。

$e(t)$，$e(nT)$ 和 $e_h(t)$ 的关系可表示如下：

$$e(t)\,|_{nT+\Delta T} = e(nT) + \frac{\mathrm{d}e}{\mathrm{d}t}\,|_{nT}\,\Delta t + \frac{\mathrm{d}e}{\mathrm{d}t^2}\,|_{nT}\,\Delta t^2 + \cdots \tag{12-10}$$

$$e(t)\,|_{nT+\Delta T} = e(nT) \qquad (0 \leqslant \Delta t < T) \tag{12-11}$$

$$e_h(t) = \sum_{k=0}^{\infty} e(kT)[1(t-(k+1)T) - 1(t-kT)] \tag{12-12}$$

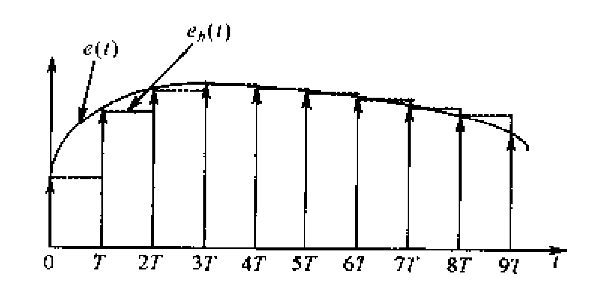


图 12.4 零阶保持器输入和输出关系图

由图 12.4 可见，零阶保持器的输出信号是阶梯信号，它与要恢复的连续信号是有区别的，包含高次谐波。若将阶梯信号的各中点连接起来，可以得到比连续信号退后 $\frac{T}{2}$ 的曲线，它这反映了零阶保持器的相位滞后特性。

零阶保持器的传递函数为：

$$E_h(s)=\sum_{k=0}^{\infty}e(kT)\mathrm{e}^{-kTs}\frac{1-\mathrm{e}^{-Ts}}{s} \tag{12-13}$$

$$E_h(s)=\frac{1-\mathrm{e}^{-Ts}}{s}E^*(s) \tag{12-14}$$

$$G_h(s)=\frac{E_h(s)}{E^*(s)}=\frac{1-\mathrm{e}^{-Ts}}{s} \tag{12-15}$$

零阶保持器的频率特性为：

$$G_h(\mathrm{j}\omega)=\frac{1-\mathrm{e}^{-\mathrm{j}T\omega}}{\mathrm{j}\omega}=T\frac{\sin(\omega T/2)}{(\omega T/2)}\mathrm{e}^{-\mathrm{j}\omega T/2} \tag{12-16}$$

零阶保持器具有如下特性。

（1）低通特性：由于幅频特性的幅值随频率值的增大而迅速衰减，说明零阶保持器基本上是一个低通滤波器，但与理想滤波器特性相比，在 $\omega=\frac{\omega_s}{2}$ 时，其幅值只有初值的 63.7%，且截止频率不止一个，所以零阶保持器除允许主要频谱分量通过外，还允许部分高频分量通过，从而造成数字控制系统的输出中存在纹波。

（2）相角特性：由相频特性可见，零阶保持器要产生相角滞后，且随着频率的增大而加大，在 $\omega=\frac{\omega_s}{2}$ 时，相角滞后可达$-180°$，从而使闭环系统的稳定性变差。

（3）时间滞后：零阶保持器的输出为阶梯信号 $e_h(t)$，其平均响应为 $e\left(t-\frac{T}{2}\right)$，表明输出比输入在时间上要滞后 $\frac{T}{2}$，相当于给系统增加一个延迟时间为 $\frac{T}{2}$ 的延迟环节，对系统稳定不利。

2. 一阶保持器

一阶保持器是一种按线性规律外推的保持器，其外推关系为：

$$e(t)|_{nT+\Delta T}=e(nT)+\frac{\mathrm{d}e}{\mathrm{d}t}|_{nT}\ \Delta t \tag{12-17}$$

由于未引进高阶差分，一阶保持器的输出信号与原连续信号之间仍有差别。一阶保持器的单位脉冲响应可以分解为阶跃函数和斜坡函数之和。

一阶保持器的单位脉冲函数的拉氏变换式可用下式表示：

$$e(t)|_{nT+\Delta T}=e(nT)+\frac{e((n+1)T)-e(nT)}{T}\Delta T \tag{12-18}$$

$$G_h(s)=\frac{E_h(s)}{E^*(s)}=T(1+Ts)\frac{1-\mathrm{e}^{-Ts}}{s} \tag{12-19}$$

12.3 Z 变 换

大多数离散控制系统可以用线性离散系统的数学模型来描述，对于线性时不变离散系统，人们习惯用线性定常系数差分方程或脉冲传递函数来表示。线性差分方程的解法主要包括迭代法、古典法和变换法，迭代法和古典法的解法比较麻烦，变换法能把复杂的计算变换成简单的代数运算。Z 变换方法就是一种常用的变换法，在求解差分方程式时，采用 Z 变换能使求解变得十分简便。

12.3.1 离散信号的 Z 变换

12.3.1.1 Z 变换定义

设连续时间函数 $f(t)$ 可进行拉氏变换，其拉氏变换为 $F(s)$。连续时间函数 $f(t)$ 经采样周期为 T 的采样开关后，变成离散信号 $f^*(t)$，用数学模型表示为：

$$f^*(t)=f(t)\sum_{k=0}^{\infty}\delta(t-kT)=\sum_{k=0}^{\infty}f(kT)\delta(t-kT) \tag{12-20}$$

对上式进行拉氏变换，得

$$F^*(s)=\sum_{k=0}^{\infty}f(kT)\mathrm{e}^{-kTs} \tag{12-21}$$

上式中各项均含有 e^{-kTs} 因子，复变函数 $-kTs$ 在指数中，且 e^{-kTs} 是超越函数，因此计算很不方便。

可令 $z=\mathrm{e}^{Ts}$，其中 T 为采样周期，z 是复数平面上定义的一个复变量，通常称为 Z 变换算子，可表示为：

$$z=\mathrm{e}^{sT}\Rightarrow s=\frac{1}{T}\ln z \tag{12-22}$$

则得到以 z 为自变量的函数 $F(z)$：

$$F(z)=\sum_{k=0}^{\infty}f(kT)z^{-k} \tag{12-23}$$

$F(z)$ 是复变量 z 的函数，它表示成一个无穷级数。如果此级数收敛，则序列的 z 变换存在。序列 $\{f(kT),k=0,1,2...\}$ 的 Z 变换存在的条件是式（12-23）所定义的级数收敛，

以及 $\lim_{N\to\infty}\sum_{k=0}^{N} f(kT)z^{-k}$ 存在。

若式（12-23）所示级数收敛，则称 $F(z)$ 是 $f^*(t)$ 的 Z 变换，记为：

$$Z(f^*(t)) = F(z) \tag{12-24}$$

$F^*(s)=\frac{1}{T}\sum_{n=-\infty}^{\infty}F[s+jn\omega_s]$ 与 $F(z)=\sum_{k=0}^{\infty}f(kT)z^{-k}$ 是相互补充的两种变换形式，前者表示 S 平面上的函数关系，后者表示 Z 平面上的函数关系。

$F(z)=\sum_{k=0}^{\infty}f(kT)z^{-k}$ 所表示的 Z 变换只适用于离散函数，或者说只能表征连续函数在采样时刻的特性，而不能反映其在采样时刻之间的特性。通常称 $F(z)$ 是 $f(t)$的 Z 变换，指的是经过采样后 $f^*(t)$ 的 Z 变换。采样函数 $f^*(t)$ 所对应的 Z 变换是惟一的，反之亦然。但是，一个离散函数 $f^*(t)$ 所对应的连续函数却不是惟一的，而是有无穷多个。从这个意义上来说，连续时间函数 $f(t)$与相应的离散时间函数 $f^*(t)$ 具有相同的 Z 变换，即

$$Z[f(t)] = Z[f^*(t)] = F(z) = \sum_{k=0}^{\infty} f(kT)z^{-k} \tag{12-25}$$

12.3.1.2　Z 变换性质定理

Z 变换有一些基本定理，可以使 Z 变换的应用变得简单方便。常用的定理有初值定理、终值定理和卷积定理。

（1）初值定理：如果 $f(t)$的 Z 变换为 $F(z)$，且 $F(z)$存在，则 $f(0)=\lim_{z\to\infty}F(z)$。

（2）终值定理：如果 $f(t)$的 Z 变换为 $F(z)$，而$(1-z^{-1})F(z)$在 Z 平面以原点为圆心的单位圆上和圆外没有极点，则 $\lim_{t\to\infty}f(t)=\lim_{t\to\infty}f(nT)=\lim_{z\to 1}(1-z^{-1})F(z)=\lim_{z\to 1}(z-1)F(z)$。

（3）卷积定理：如果 $x_c(nT)=\sum_{i=0}^{n}g[(n-i)T]x_r(iT)$，其中 $n=0, 1, 2\cdots$为正整数，且满足：当 $n=-1,-2,\ldots$时，$x_c(nT)=0$，$g(nT)=0$，$x_r(nT)=0$，则卷积定理可表示为：

$$x_c(z) = W(z)x_r(z) \tag{12-26}$$

其中，$W(z)=Z\{g(nT)\}$，$X_r(z)=Z\{x_r(nT)\}$。

常用的 Z 变换性质有线性性质、滞后性质、超前性质和位移性质，如表 12.4 所示。

表 12.4　拉氏变换常用性质

性质 \ 函数	原函数 $f(t)$	象函数 $F(z)$
线性性质	$af_1(t)+bf_2(t)$，a,b 为实数	$aF_1(z)+bF_2(z)$
滞后性质	$f(t-nT)$	$z^{-n}F(z)\ (T\geqslant 0)$
超前性质	$f(t+nT)$	$z^nF(z)-z^n\sum_{m=0}^{n-1}f(mT)z^{-m}$
位移性质	$e^{\pm at}f(t)$	$F(ze^{\mp aT})$

12.3.2 Z 变换、Z 反变换常用方法及 MATLAB 实现

12.3.2.1 Z 变换常用方法

求离散时间函数 Z 变换有多种方法，下面讲述三种常用的 Z 变换方法。

1. 部分分式法

设连续函数 $f(t)$的拉氏变换式为有理函数 $F(s)$，$F(s)$可以展开成部分分式的形式：

$$F(s)=\sum_{i=1}^{n}\frac{A_i}{s-p_i} \tag{12-27}$$

式中，p_i 为 $F(s)$ 的极点，A_i 为常系数。

$\dfrac{A_i}{s-p_i}$ 对应的时间函数为 $A_i e^{p_i t}$，其 Z 变换为：

$$Z[A_i e^{p_i t}]=A_i\frac{z}{z-e^{p_i T}} \tag{12-28}$$

则 $f(t)$的 Z 变换为：

$$F(z)=\sum_{i=1}^{n}A_i\frac{Z}{Z-e^{p_i T}} \tag{12-29}$$

利用部分分式法做 Z 变换时，应先求出已知连续时间函数 $f(t)$的拉氏变换 $F(s)$，然后将有理分式函数 $F(s)$展开成部分分式之和的形式，最后求出（或查表得到）每一项相应的 Z 变换。

2. 级数求和法

根据 Z 变换定义，由离散函数 $f^*(t)$：

$$f^*(t)=f(t)\sum_{k=0}^{\infty}\delta(t-kT)=\sum_{k=0}^{\infty}f(kT)\delta(t-kT) \tag{12-30}$$

及其反变换 $F^*(s)$：

$$F^*(s)=\sum_{k=0}^{\infty}f(kT)e^{-kTs} \tag{12-31}$$

可得：

$$F(z)=\sum_{k=0}^{\infty}f(kT)z^{-k}=f(0)+f(T)z^{-1}+f(2T)z^{-2}+\cdots+f(kT)z^{-k}+\cdots \tag{12-32}$$

式（12-32）是离散函数 Z 变换的一种表达形式。只要已知连续函数在采样时刻 $kT(k=0,1,2,\cdots)$ 的采样值，便可求取离散函数 Z 变换的级数展开式。对常用离散函数的 Z 变换应写成级数的求和形式。

通过级数求和法求取已知函数 Z 变换的缺点在于需要将无穷级数写成和式。这在某些情况下要求具有很高的技巧。但函数 Z 变换的无穷级数形式具有鲜明的物理含义，这

又是 Z 变换无穷级数表达形式的优点。Z 变换本身便包含着时间概念，可由函数 Z 变换的无概数形式概楚地看出原连续函数采样脉冲序列的分布情况。

3. 留数法

如果已知连续信号$f(t)$的拉氏变换$F(s)$及它的全部极点，则可用下列留数计算公式求取$F(z)$：

$$F(z)=\sum_{i=1}^{n}\underset{s=s_i}{\text{Res}}[F(s)\frac{z}{z-e^{Ts}}] \tag{12-33}$$

函数$F(s)\frac{z}{z-e^{Ts}}$在概点处的留数计算方法如下：

若s_i为单极点，则：

$$\text{Res}[F(s)\frac{z}{z-e^{Ts}}]_{s\to s_i}=\lim_{s\to s_i}[(s-s_i)F(s)\frac{z}{z-e^{Ts}}] \tag{12-34}$$

若$F(s)\frac{z}{z-e^{Ts}}$有r_i重概点s_i，则：

$$\text{Res}[F(s)\frac{z}{z-e^{sT}}]_{s\to s_i}=\frac{1}{(r_i-1)!}\lim_{s\to s_i}\frac{d^{r_i-1}[(s-s_i)^{r_i}F(s)\frac{z}{z-e^{sT}}]}{ds^{r_i-1}} \tag{12-35}$$

12.3.2.2　Z 反变换常用方法

与拉氏反变换类似，Z 反变换可表示为：

$$Z^{-1}[(F(z)]=f(kT) \tag{12-36}$$

下面介绍三种常用的 Z 反变换法。

1. 综合除法

这种方法是用$F(z)$的分母除分子。求出z^{-1}按升幂排列的级数展开式，然后用反变换求出相应的采样函数的脉冲序列。

$F(z)$可表示为：

$$F(z)=\frac{b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m}}{1+a_1z^{-1}+a_2z^{-2}+\cdots+a_nz^{-n}}\qquad(m\leqslant n) \tag{12-37}$$

其中a_i，b_j均为常系数。

通过对式（12-37）直接做综合除法，可得按z^{-1}升幂排列的幂级数展开式。如果得到的无穷级数是收敛的，则概 Z 变换定义可知上式中的系数$f_k(k=0,1,\cdots)$就是采样脉冲序列$f^*(t)$的脉冲强度$f(kT)$。

这样就可直接写出$f^*(t)$的脉冲序列表达式为：

$$f^*(t)=\sum_{k=0}^{\infty}f_k\delta(t-kT) \tag{12-38}$$

上式就是要求通过 Z 反变换得到的离散信号$f^*(t)$。

使用综合除法有几点要注意：

（1）在进行综合除法之前，必须先将 $F(z)$ 的分子、分母多项式按 z 的降幂形式排列。

（2）实际应用中，常常只需计算有限的几项就够了，因此用综合除法计算 $f^*(t)$ 最简便，这是它的优点之一。

（3）要从一组 $f(kT)$ 值中求出通项表达式，通常是比较困难的。

2. 部分分式展开法

在 Z 变换表中，所有 Z 变换函数 $F(z)$ 在其分子上都普遍含有因子 z，所以应将 $\frac{F(z)}{z}$ 展开为部分分式，然后将所得结果每一项都乘以 z，即得 $F(z)$ 的部分分式展开式。

3. 留数计算法

根据 z 变换定义有：

$$F(z)=\sum_{k=0}^{\infty}f(kT)z^{-k} \tag{12-39}$$

根据柯西留数定理有：

$$f(kT)=\sum_{i=1}^{n}\mathrm{Res}[F(z)z^{k-1}]_{z\to z_i} \tag{12-40}$$

式中 $\mathrm{Res}[F(z)z^{k-1}]_{z\to z_i}$ 表示 $F(z)z^{k-1}$ 在极点 z_i 处的留数。

函数 $F(z)z^{k-1}$ 在极点处的留数计算方法如下：

若 z_i 为单极点，则：

$$\mathrm{Res}[F(z)z^{k-1}]_{z\to z_i}=\lim_{z\to z_i}[(z-z_i)F(z)z^{k-1}] \tag{12-41}$$

若 $F(z)z^{k-1}$ 有 r_i 阶重极点，则：

$$\mathrm{Res}[F(z)z^{k-1}]_{z\to z_i}=\frac{1}{(r_i-1)!}\lim_{z\to z_i}\frac{\mathrm{d}^{r_i-1}[(z-z_i)^{r_i}F(z)z^{k-1}]}{\mathrm{d}z^{r_i-1}} \tag{12-42}$$

常用时间函数的 Z 变换如表 12.5 所示，供在进行相应函数 Z 变换时查阅。

表 12.5　常用函数 Z 变换表

时域函数 $y(t)$	拉氏变换函数 $Y(s)$	Z 变换函数 $Y(z)$
$\delta(t)$	1	1
$\delta(t-kT)$	e^{-kTs}	z^{-k}
$1(t)$	$\frac{1}{s}$	$\frac{z}{z-1}$
t	$\frac{1}{s^2}$	$\frac{Tz}{(z-1)^2}$
$\frac{t^2}{2}$	$\frac{1}{s^3}$	$\frac{T^2z(z+1)}{2(z-1)^3}$
$\frac{t^3}{3!}$	$\frac{1}{s^4}$	$\frac{T^3(z^2+4z+1)}{6(z-1)^4}$

续表

时域函数 $y(t)$	拉氏变换函数 $Y(s)$	Z 变换函数 $Y(z)$
e^{-at}	$\frac{1}{s+a}$	$\frac{z}{z-e^{-aT}}$
te^{-at}	$\frac{1}{(s+a)^2}$	$\frac{Te^{-aT}z}{(z-e^{-aT})^2}$
$1-e^{-at}$	$\frac{a}{s(s+a)}$	$\frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}$
$\sin(\omega t)$	$\frac{\omega}{s^2+\omega^2}$	$\frac{z\sin(\omega T)}{z^2-2z\cos(\omega T)+1}$
$\cos(\omega t)$	$\frac{s}{s^2+\omega^2}$	$\frac{z^2-z\cos(\omega T)}{z^2-2z\cos(\omega T)+1}$
$e^{-at}\sin(\omega t)$	$\frac{\omega}{(s+a)^2+\omega^2}$	$\frac{ze^{-aT}\sin(\omega T)}{z^2-2ze^{-aT}\cos(\omega T)+e^{-2aT}}$
$e^{-at}\cos(\omega t)$	$\frac{s+a}{(s+a)^2+\omega^2}$	$\frac{z^2-ze^{-aT}\cos(\omega T)}{z^2-2ze^{-aT}\cos(\omega T)+e^{-2aT}}$

12.3.2.3　Z 变换和 Z 反变换的 MATLAB 实现

MATLAB 提供了符号运算工具箱（Symbolic Math Toolbox），可方便地进行 Z 变换和 Z 反变换，进行 Z 变换的函数是 ztrans，进行 Z 反变换的函数是 iztrans。

1．ztrans

函数调用格式如下。

F=ztrans(f)：函数返回独立变量 n 关于符号向量 f 的 Z 变换函数：ztrans(f) ⇔ F(z)=symsum(f(n)/z^n,n,0,inf)，这是默认的调用格式。

F=ztrans(f, w)：函数返回独立变量 n 关于符号向量 f 的 Z 变换函数，只是用 w 代替了默认的 z：ztrans(f,w) ⇔ F(w)=symsum(f(n)/w^n,n,0,inf)。

F=ztrans(f, k, w)：函数返回独立变量n关于符号向量k的Z变换函数：ztrans(f,k,w) ⇔ F(w)= symsum(f(k)/w^k,k,0,inf)。

2．iztrans

函数调用格式如下。

f=iztrans(F)：函数返回独立变量 z 关于符号向量 F 的 Z 反变换函数，这是默认的调用格式。

f=iztrans(F, k)：函数返回独立变量 k 关于符号向量 F 的 Z 反变换函数，只是用 k 代替了默认的 z。

f=iztrans(F, w, k)：函数返回独立变量 w 关于符号向量 F 的 Z 反变换函数。

【例 12-1】　试求下列函数的 Z 变换。

（1）$f_1(t)=t$

（2）$f_2(t)=e^{-at}$

（3） $f_3(t)=\sin(at)$

解：使用 MATLAB 提供的符号工具箱函数进行计算，程序代码如下。

```
%创建符号变量，T 为采样周期
syms n a w k z T
%进行函数f1(t)=t 的 z 变换
x1=ztrans(n*T)
%化简结果
x1=simplify(x1)
%进行函数f2(t)=e^-at 的 z 变换
x2=ztrans(exp(-a*n*T))
%化简结果
x2=simplify(x2)
%进行函数f3(t)=sin(at)的 z 变换
x3=ztrans(sin(w*a*T), w, z)
%化简结果
x3=simplify(x3)
```

运行结果如下：

```
x1 =
T*z/(z-1)^2
x1 =
T*z/(z-1)^2
x2 =
z/exp(-a*T)/(z/exp(-a*T)-1)
x2 =
z*exp(a*T)/(z*exp(a*T)-1)
x3 =
z*sin(a*T)/(z^2-2*z*cos(a*T)+1)
x3 =
z*sin(a*T)/(z^2-2*z*cos(a*T)+1)
```

可见，变换结果为：

$$F_1(z)=\frac{Tz^{-1}}{(1-z^{-1})^2}$$

$$F_2(z)=\frac{1}{1-e^{-aT}z^{-1}}$$

$$F_3(z)=\frac{\sin(aT)z^{-1}}{1-2(\cos(aT)z^{-1}+z^{-2}}$$

【例 12-2】　试求下列函数的 Z 变换。

（1） $F_1(s)=\dfrac{1}{s(s+1)}$

（2） $F_2(s)=\dfrac{s}{s^2+a^2}$

（3） $F_3(s)=\dfrac{a-b}{(s+a)(s+b)}$

解：使用 MATLAB 提供的符号工具箱函数进行计算，由于只有时域的 Z 变换，因此对于拉氏域首先需要进行拉氏反变换，程序代码如下。

```
%创建符号变量，T 为采样周期
syms s n t1 t2 t3 a b k z T
%进行函数 F1(s)的拉氏反变换
x1= ilaplace(1/s/(s+1), t1)
%化简结果
x1=simplify(x1)
%进行函数 F2(s)的拉氏反变换
x2= ilaplace( s/(s^2+a^2), t2)
%化简结果
x2=simplify(x2)
%进行函数 F3(s)的拉氏反变换
x3= ilaplace( (a-b)/(s+a)/(s+b), t3)
%化简结果
x3=simplify(x3)
```

运行结果如下：

```
x1 =
1-exp(-t1)
x1 =
1-exp(-t1)
x2 =
cos((a^2)^(1/2)*t2)
x2 =
cos(csgn(a)*a*t2)
x3 =
(a-b)/(b-a)*(exp(-a*t3)-exp(-b*t3))
x3 =
-exp(-a*t3)+exp(-b*t3)
```

对拉氏反变换结果进行 Z 变换，注意把时间参数 t1, t2, t3 都替换成 n*T，在命令窗口中运行，过程如下：

```
x1=ztrans(1-exp(-n*T))
x1=simplify(x1)
x1 =
z/(z-1)-z/exp(-T)/(z/exp(-T)-1)
x1 =
z*(-1+exp(T))/(z-1)/(z*exp(T)-1)
>> x2=ztrans(cos((a^2)^(1/2)*n*T))
x2=simplify(x2)
x2 =
(z-cos(signum(a)*a*T))*z/(z^2-2*z*cos(signum(a)*a*T)+1)
x2 =
(z-cos(signnm(a)*a*T))*z/(z^2-2*z*cos(signum(a)*a*T)+1)
% signum(a)求 a 的符号
>>  x3=ztrans(-exp(-a*n*T)+exp(-b*n*T))
x3=simplify(x3)
x3 =
-z/exp(-a*T)/(z/exp(-a*T)-1)+z/exp(-b*T)/(z/exp(-b*T)-1)
x3 =
-z*(-exp(a*T)+exp(b*T))/(z*exp(a*T)-1)/(z*exp(b*T)-1)
```

可见，变换结果为：

$$F_1(z)=\frac{z}{z-1}-\frac{z}{z-\mathrm{e}^{-T}}=\frac{z(1-\mathrm{e}^{-T})}{(z-1)(z-\mathrm{e}^{-T})}$$

$$F_2(z)=\frac{1-\cos(aT)z^{-1}}{1-2(\cos(aT)z^{-1}+z^{-2}}$$

$$F_3(z)=\frac{(\mathrm{e}^{-bT}-\mathrm{e}^{-aT})z^{-1}}{(1-\mathrm{e}^{-aT}z^{-1})(1-\mathrm{e}^{-bT}z^{-1})}$$

【例 12-3】 试求下列函数的 Z 反变换。

（1） $F_1(z)=\dfrac{2z^2-0.5z}{z^2-0.5z-0.5}$

（2） $F_2(z)=\dfrac{z+0.5}{z^2+3z+2}$

解：使用 MATLAB 提供的符号工具箱函数进行计算，程序代码如下。

```
%创建符号变量，T 为采样周期
syms z a k T
%进行函数 F1(z)的 z 反变换
x1=iztrans( (2*z^2-0.5*z)/(z^2-0.5*z-0.5))
%化简结果
x1=simplify(x1)
%进行函数 F2(z)的 z 反变换
```

```
x2=iztrans( (z+0.5)/(z^2+3*z+2) )
%化简结果
x2=simplify(x3)
```

运行程序，输出拉氏反变换结果如下：

```
x1 =
(-1/2)^n+1
x1 =
(-1)^n*2^(-n)+1
x2 =
1/2*(-1)^n-3/4*(-2)^n
x2 =
1/2*(-1)^n+3/4*(-1)^(1+n)*2^n
```

可见，变换结果为：

$$f_1(kT)=\sum_{k=0}^{\infty}f(kT)\delta(t-kT)\text{，}\quad f_2(kT)=0.5(-1)^k-0.75(-2)^k$$

12.4　离散控制系统数学模型

要对一个线性离散系统或近似线性离散系统进行分析和设计，首先需要建立相应的系统模型，解决数学描述和分析工具问题。

可用时域数学模型和频域数学模型对线性离散系统进行描述，频域数学模型主要是差分方程，频域数学模型主要是脉冲传递函数。

12.4.1　离散系统时域数学模型

对于一般的线性定常离散系统，如图 12.5 所示，k 时刻的输出 $y(k)$不但与 k 时刻的输入 $x(k)$有关，而且与 k 时刻以前的输入 $x(k-1)$, $x(k-2)\cdots$有关，同时还与 k 时刻以前的输出 $y(k-1)$, $y(k-2)\cdots$有关。这种关系可以用下列差分方程描述：

r(kT) → f[r(kT)] → y(kT)

图 12.5　线性离散系统模型

$$\begin{aligned}&y(kT)+a_1y(kT-T)+a_2y(kT-2T)+\ldots+a_{n-1}y(kT-nT+T)+a_ny(kT-nT)\\&=b_0r(kT)+b_1r(kT-T)+b_2r(kT-2T)+\ldots+b_{m-1}r(kT-mT+T)+b_mr(kT-mT)\end{aligned}\tag{12-43}$$

上式可表示为：

$$y(kT)=\sum_{i=0}^{m}b_ir(kT-iT)-\sum_{i=1}^{n}a_iy(kT-iT)\tag{12-44}$$

式中，a 和 b 为常数，$m<n$, $k=0, 1, 2, \cdots$，称为阶线性常系数差分方程，它在数学上代表一个线性定常离散系统。

差分方程的解法常用方法有迭代法和 Z 变换法。迭代法非常简单，此处不再赘述。

Z 变换法的实质是利用 Z 变换的实数位移定理，将差分方程化为以 z 为变量的代数

方程，然后进行 Z 反变换，求出各采样时刻的响应。

Z 变换法的具体步骤是：

（1）对差分方程进行 Z 变换；

（2）解出方程中输出量的 Z 变换 $Y(z)$；

（3）求 $Y(z)$的 Z 反变换，得差分方程的解 $y(k)$。

【例 12-4】 已知一个离散线性系统的差分方程描述如下：

$$y(k+3)-2.7y(k+2)+2.42y(k+1)-0.72y(k)$$
$$=0.1r(k+2)+0.03r(k+1)-0.07r(k)$$

试建立系统的传递函数，显示对象的属性，提取分子和分母多项式，并提取零极点和增益。

解：在零初始条件下，对差分方程进行 Z 变换，得到系统的传递函数：

$$G(z)=\frac{0.1z^2+0.03z-0.07}{z^3-2.7z^2+2.42z-0.72}$$

由于系统未指明采样周期，使用命令 tf 时，它的第 3 个参数应选为‘–1’，表示采样周期 T 未知；如果 T 已知，第 3 个参数则用 T 值来代替；如果第 3 个参数完全省略了，则表示系统是连续的或非离散的。

MATLAB 程序代码如下：

```
% 差分方程输入变量的系数
numG = [0.1 0.03 -0.07]
% 差分方程输出变量的系数
denG = [1 -2.7 2.42 -0.72]
%建立传递函数模型
G = tf(numG, denG, -1)
% 获得模型属性
get(G)
%提取分子和分母多项式
[nn, dd] = tfdata(G, 'v')
%提取对象的零极点和增差
[zz, pp, kk] = zpkdata(G, 'v')
%画零极点图
pzmap(G)
```

MATLAB 命令将建立 $G(z)$作为传递函数对象，显示其所有的当前属性，创建两个包括分子、分母多项式系统的列向量 nn 和 dd，并画出零极点图形显示于图 12.6 中，系统的极点是 $p_{1,2,3}=0.8,0.9,1.0$，零点是 $z_{1,2}=0.7,-1.0$，增益是 0.1。

程序运行结果如下：

```
Transfer function:
    0.1 z^2 + 0.03 z - 0.07
-----------------------------
z^3 - 2.7 z^2 + 2.42 z - 0.72
%对象属性
Sampling time: unspecified
             num: {[0 0.1 0.03 -0.07]}
             den: {[1 -2.7 2.42 -0.72]}
        Variable: 'z'
              Ts: -1
         ioDelay: 0
      InputDelay: 0
     OutputDelay: 0
       InputName: {''}
      OutputName: {''}
      InputGroup: {0x2 cell}
     OutputGroup: {0x2 cell}
           Notes: {}
        UserData: []
%分子和分母多项式
nn =
          0    0.1000    0.0300   -0.0700
dd =
     1.0000   -2.7000    2.4200   -0.7200
%系统的零极点和增益
zz =
    -1.0000
     0.7000
pp =
     1.0000
     0.9000
     0.8000
kk =
     0.1000
```

输出图形如图 12.6 所示。

12.4.2 离散系统统统数学模型

在线性连续系统中，将初始条件为零时系统（或环节）输出信号的拉氏变换与输入信号的拉氏变拉之比定义为传递函数，并用它来描述系统（或环节）的特性。

与此相似，在线性离散系统中，将初始条件为零时系统（或环节）的输出信号 Z 变换与输入信号的 Z 变换之比定义为脉冲传递函数，又称为 Z 传递函数。脉冲传递函数是

离散系统的一个重要概念，是分析离散系统的有力工具。

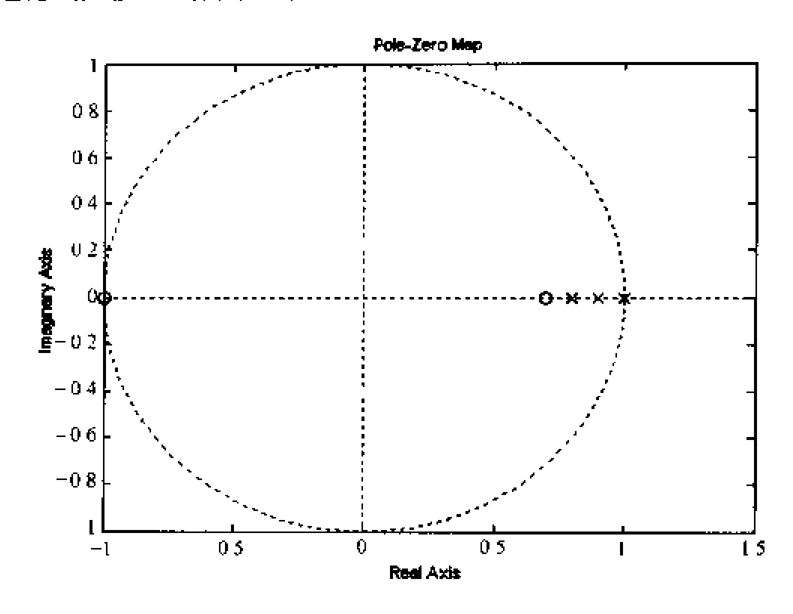


图 12.6　例 12-4 的零极点图

12.4.2.1　脉冲传递函数定义

在零初始条件下，线性定常离散系统的离散输出信号 Z 变换与离散输入信号 Z 变换之比称为该系统的脉冲传递函数（或 Z 传递函数）。

$$G(z)=\frac{Y(z)}{X(z)} \tag{12-45}$$

应该指出，多数实际采样系统的输出信号是连续信号，如图 12.7 所示。在这种情况下，可以在输出端虚设一个采样开关，并设它与输入采样开关以相同的采样周期 T 同步工作，这样就可以沿用脉冲传递函数的概念。

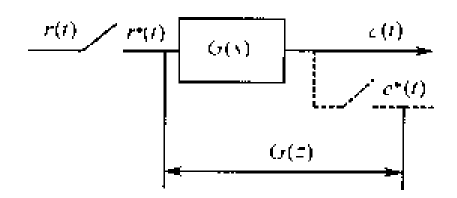


图 12.7　采样系统脉冲传递函数

现分析一个孤立的单位脉冲函数 $\delta(t)$ 加在线性对象 $G(s)$ 上的情况。

由于 $\delta(t)$ 的拉氏变换等于 1，所以输出量的拉氏变换为 $y(s)=G(s)$，进一步有 $y(s)=L^{-1}[G(s)]$。习惯上将脉冲响应函数用 $g(t)$ 表示，即 $g(t)=L^{-1}[G(s)]$。

如果在 $G(s)$ 上加的是 $\delta(t-a)$，即延迟到 a 时刻才将脉冲函数加上，那么输出信号也自然地延迟一段时间 a，而成为 $g(t-a)$。

再研究一系列脉冲依次作用到 $G(s)$ 上的情况。

脉冲序列 $x^*(t)$ 可以表示成：

$$x^*(t)=x(0)\delta(t)+x(T)\delta(t-T)+x(2T)\delta(t-2T)+\cdots \tag{12-46}$$

为了求解输出量在各个采样时刻的值，先计算各段时间内的 $y(t)$。

在 $0 \leqslant t < T$ 时间内，实际起作用的只有 $t=0$ 时刻加入的那一个脉冲，其余各个脉冲尚未加入。因此在这段时间内的输出量是 $y(t)=x(0)g(t)$，将 $t=0$ 代入上式得 $y(0)=x(0)g(0)$。

在 $T \leqslant t < 2T$ 时间内，$y(t)=x(0)g(t)+x(T)g(t-T)$，将 $t=T$ 代入上式得 $y(T)=x(0)g(t)+\ x(T)g(0)$，以此类推，可得出输出在各个采样时刻的值 $y(kT),(k=0,1,2,\cdots$。

于是 $y(t)$的 Z 变换为：

$$
\begin{aligned}
Y(z)&=\sum_{k=0}^{\infty} y(kT)z^{\ k}=y(0)z^{0}+y(T)z^{-1}+y(2T)z^{-2}+\cdots \\
&=x(0)g(0)+[x(0)g(T)+x(T)g(0)]z^{-1} \\
&\quad +[x(0)g(2T)+x(T)g(T)+x(2T)g(0)]z^{-2}+\cdots \\
&=x(0)[g(0)+g(T)z^{-1}+g(2T)z^{-2}+\cdots] \\
&\quad +x(T)[g(0)z^{-1}+g(T)z^{-2}+g(2t)z^{-3}+\cdots] \\
&\quad +x(2T)[g(0)z^{-2}+g(T)z^{-3}+\cdots]+\cdots \\
&=x(0)[g(0)+g(T)z^{-1}+g(2T)z^{-2}+\cdots] \\
&\quad +x(T)z^{-1}[g(0)+g(T)z^{-1}+g(2T)z^{-2}+\cdots] \\
&\quad +x(2T)z^{-2}[g(0)+g(T)z^{-1}+\cdots]+\cdots \\
&=[g(0)+g(T)z^{-1}+g(2T)z^{-2}+\cdots][x(0)+x(T)z^{-1}+x(2T)z^{-2}+\cdots] \\
&=\sum_{k=0}^{\infty} g(kT)z^{-k}\cdot\sum_{k=0}^{\infty} x(kT)z^{-k} \\
&=G(z)X(z)
\end{aligned}
\tag{12-47}
$$

12.4.2.2　脉冲传递函数求解

连续系统或元件的脉冲传递函数 $G(z)$可以通过其传递函数 $G(s)$来求取。

方法是：先求 $G(s)$的拉氏反变换，得到脉冲过渡函数 $g(t)$，再将 $g(t)$按采样周期离散化，得到加权序列 $g(nT)$，最后将 $g(nT)$进行 Z 变换，得出 $G(z)$。这一过程比较复杂，通常可根据 Z 变换表，直接从 $G(s)$得到 $G(z)$，而不必逐步推导。

若已知系统的差分方程，则可对方程两端进行 Z 变换，应用 $G(z)=\dfrac{Y(z)}{X(z)}$ 求解。

1. 采样拉氏变换的两个重要性质

（1）采样函数的拉氏变换具有周期性，即 $G^{*}(s)=G^{*}(s+\mathrm{j}k\omega_{s})$。

（2）若采样函数的拉氏变换与连续函数的拉氏变换相乘后再离散化，则可以从离散符号中提出来，即 $[G(s)E^{*}(s)]^{*}=G^{*}(s)E^{*}(s)$。

2. 开环脉冲传递函数

采样系统在开环状态下有两种不同的结构形式，分别如图 12.8 和图 12.9 所示。

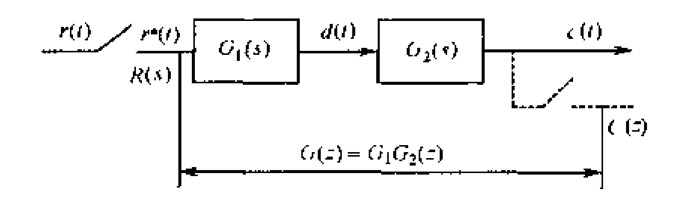


图 12.8　中间没有采样开关的两个环节串联

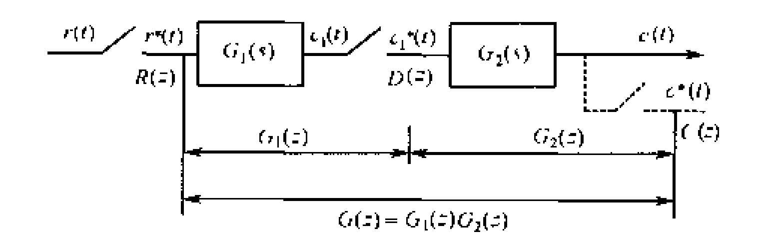


图 12.9　中间有采样开关的两个环节串联

串联环节之间无采样器时的脉冲传递函数为：

$$G(z)=\frac{Y(z)}{X(z)}=G_1G_2(z) \tag{12-48}$$

串联环节之间有采样器时的脉冲传递函数为：

$$G(z)=\frac{Y(z)}{X(z)}=G_1(z)G_2(z) \tag{12-49}$$

上式表明，被采样开关分隔的两个线性环节串联时，其脉冲传递函数等于这两环节的脉冲传递函数之积。这个结论可以推广到有 n 个环节串联而各相邻环节之间都有采样开关分离的情形。无采样开关分隔的两个线性环节串联时，其脉冲传递函数等于这两个环节传递函数之积的 Z 变换。显然，这一结论也可以推广到有 n 个环节直接串联的情况，但环节之间存在与不存在采样开关时的脉冲传递函数是不相同的。

3. 带有零阶保持器的开环系统脉冲传递函数

设有如图 12.10 所示的带零阶保持器的开环系统，经简单变换为如图 12.11 所示的等效的开环系统。

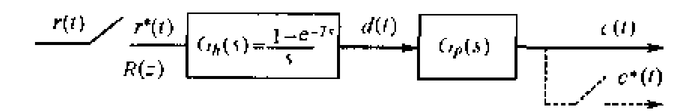


图 12.10　带零阶保持器的开环系统

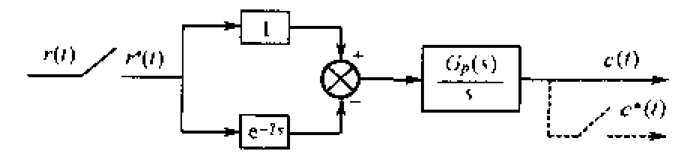


图 12.11　带有零阶保持器的开环系统等效图

根据实数位移定理及采样拉氏变换性质，可得：

$$Y(s)=[\frac{G_p(s)}{s}-e^{-sT}\frac{G_p(s)}{s}]X^*(s) \tag{12-50}$$

其 Z 变换式如下：

$$Y(z)=Z\left[\frac{G_p(s)}{s}\right]X(z)-z^{-1}Z\left[\frac{G_p(s)}{s}\right]X(z) \tag{12-51}$$

于是，在有零阶保持器时，开环系统脉冲传递函数为：

$$G(z)=\frac{Y(z)}{X(z)}=(1-z^{-1})Z\left[\frac{G_p(s)}{s}\right] \tag{12-52}$$

4. 采样系统的闭环脉冲传递函数

在采样系统中，由于设置采样器方式是多种多样的，所以闭环系统的结构形式也不是统一的。图 12.12 是比较常见的系统结构图，图中输入端和输出端的采样开关是为了便于分析而虚设的。

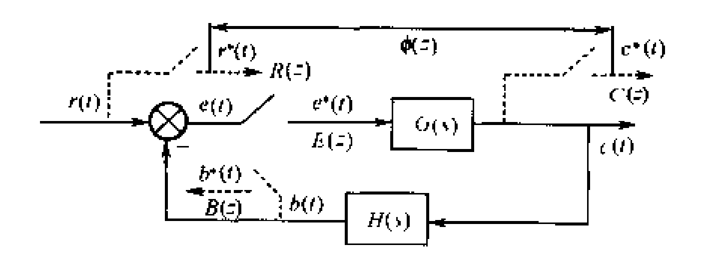


图 12.12　采样闭环系统结构示意图

系统闭环脉冲传递函数为：

$$\Phi(z)=\frac{Y(z)}{R(z)}=\frac{G(z)}{1+HG(z)} \tag{12-53}$$

系统闭环误差脉冲传递函数为：

$$\Phi_e(z)=\frac{E(z)}{R(z)}=\frac{1}{1+HG(z)} \tag{12-54}$$

与连续系统类似，令 $\Phi(z)$ 或 $\Phi_e(z)$ 的分母多项式为零，便可得到离散系统的特征方程如下：

$$D(z)=1+HG(z)=0 \tag{12-55}$$

需要指出，离散闭环系统脉冲传递函数不能从 $\Phi(z)$ 和 $\Phi_e(z)$ 求 Z 变换得来，即

$$\Phi(z)\neq Z[\Phi(s)],\ \Phi_e(z)\neq Z[\Phi_e(s)] \tag{12-56}$$

通过与上面类似的方法可以导出采样器为不同配置形式的其他闭环系统脉冲传递函数。但只要误差信号 $e(t)$ 处没有采样开关，则输入采样信号 $r^*(t)$ 就不存在，此时不能写出闭环系统对于输入量的脉冲传递函数，而只能求出输出采样信号的 Z 变换函数 $Y(z)$ 或 $C(z)$。

对于采样开关在闭环系统中具有各种配置的闭环离散系统结构图，以及其输出采样信号 Z 变换函数 $Y(z)$，此处不再赘述。

MATLAB 提供了连续系统和离散系统相互转换的函数，如表 12.6 所示。

表 12.6 连续系统模型与离散系统模型转换函数

函　数	调用格式	函数说明
c2d	sysd=c2d（sysc, Ts, 'method'）	连续时间 LTI 系统模型转换成离散时间系统模型
c2dm	[Ad, Bd, Cd, Dd]=2dm（A, B, C, D, Ts, 'method'），[numd, dend]=c2dm（num, den, Ts, 'method'）	连续时间 LTI 系统状态空间模型或传递函数模型转换或离数时间系统模型
d2c	sysc=d2c（sysd, , 'method'）	离数时间 LTI 系统模型转换成连续时间系统模型
d2cm	[A, B, C, D]=d2cm（Ad, Bd, Cd, Dd, Ts, 'method'）	离数时间 LTI 系统模型转换成连续时间系统模型
d2d	Sys=d2d（sysd, Ts）	离散时间系统模型转换成新的 Ts 离散时间系统模型
d2dt	[Ad, Bd, Cd, Dd]=c2dt（A, B, C, Ts, lambda）	具有纯延迟 lambda 输入的连续时间 LTI 状态空间系统转换成离散时间状态空间系统

在表 12.6 中，d 表示离散系统（discrete）；c 表示连续系统（continuous）；2 表示 to（转换成的含义，在其他函数中也经常这样使用）；Ts 表示采样周期，单位为 s。

表 12.6 函数中的'method'表示转换是选用的变换方法，基本含义如表 12.7 所示。

表 12.7 选项'method'的功能说明

选　项	功能说明
'zoh'	时输入信号加零阶保持器
'foh'	对输入信号加一阶保持器
'imp'	脉冲不变变换方法
'tustin'	双线性变换方法
'prewarp'	预先转折变换方法，即改进的双线性变换方法
'matched'	零极点匹配变换方法

其中，默认的方式是'zoh'。

【例 12-5】 已知一个连续线性系统如图 12.13 所示，对象模型 $G_p(s)=\frac{1}{s(s+1)}$，试用零阶保持器方法、一阶保持器方法、双线性变换方法和根匹配方法将此连续系统离散化，其中采样周期 $T_s=0.1\text{s}$。

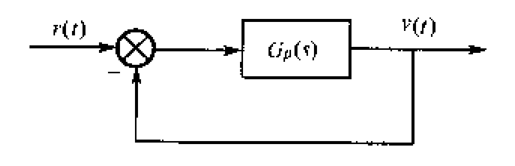


图 12.13 例 12-5 的连续系统结构图

解：MATLAB 程序代码如下。

```
num=[1]
den=[1 1 0]
%采样周期
Ts=0.1
%建立连续系统传递函数模型
Gs=tf(num, den)
%采用零阶保持方法进行系统变换
Gd1=c2d(Gs, Ts, 'zoh')
%采用一阶保持方法进行系统变换
Gd2=c2d(Gs, Ts, 'foh')
%采用双线性变换方法进行系统变换
Gd3=c2d(Gs, Ts, 'tustin')
%采用零极点匹配变换方法进行系统变换
Gd4=c2d(Gs, Ts, 'matched')
```

运行结果如下：

```
%零阶保持方法
Transfer function:
0.004837 z + 0.004679
---------------------
z^2 - 1.905 z + 0.9048
Sampling time: 0.1
%一阶保持方法
Transfer function:
0.001626 z^2 + 0.006344 z + 0.001547
-----------------------------------
          z^2 - 1.905 z + 0.9048
Sampling time: 0.1
%双线性变换方法
Transfer function:
0.002381 z^2 + 0.004762 z + 0.002381
-----------------------------------
          z^2 - 1.905 z + 0.9048
Sampling time: 0.1
%极匹配变换方法
Transfer function:
0.005004 z + 0.005004
---------------------
z^2 - 1.905 z + 0.9048
```

```
Sampling time: 0.1
```

【例 12-6】 已知一个连续线性系统如图 12.14 所示，对象模型 $G_1(s)=\dfrac{2}{s(s+30)}$，$G_2(s)=\dfrac{10}{s^2+6s+5}$，采样周期 $T_s=0.1\text{s}$，试求系统的脉冲闭环传递函数。

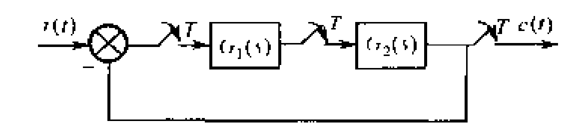


图 12.14　例 12-6 系统结构图

解：MATLAB 程序代码如下。

```
Clear
%采样周期
Ts=0.1
num1=[2]
den1=[1, 30, 0]
num2=[10]
den2=[1, 6, 5]
G1c=tf(num1, den1)
G2c=tf(num2, den2)
%采用零阶保持方法进行系统变换
G1d=c2d(G1c, Ts)
%采用零阶保持方法进行系统变换
G2d=c2d(G2c, Ts)
Gd=G1d*G2d
%建立闭环系统模型
GHd=feedback(Gd, 1)
```

运行结果如下：

```
%G1(s)的传递函数
Transfer function:
     2
----------
s^2 + 30 s
%G2(s)的传递函数
Transfer function:
       10
-------------
s^2 + 6 s + 5
%G1(s)转换后的 z 传递函数
```

```
Transfer function:
 0.004555 z + 0.00178
----------------------
z^2 - 1.05 z + 0.04979
Sampling time: 0.1
%G2(s)转换后的 z 传递函数
Transfer function:
 0.04117 z + 0.03372
----------------------
z^2 - 1.511 z + 0.5488
Sampling time: 0.1
%开环系统的 z 传递函数
Transfer function:
        0.0001875 z^2 + 0.0002268 z + 6e-005
-----------------------------------------------
z^4 - 2.561 z^3 + 2.185 z^2 - 0.6514 z + 0.02732
Sampling time: 0.1
%闭环系统的 z 传递函数
Transfer function:
        0.0001875 z^2 + 0.0002268 z + 6e-005
-----------------------------------------------
z^4 - 2.561 z^3 + 2.185 z^2 - 0.6512 z + 0.02738
Sampling time: 0.1
```

12.5　离散控制系统分析

12.5.1　离散控制系统的稳定性

在线性连续系统中，根据特征方程的根在 S 平面的位置判别系统的稳定性。若系统特征方程的所有根都在 S 平面左半平面，则系统稳定。对线性离散系统进行了 Z 变换以后，对系统的分析要采用 Z 平面，因此需要弄清这两个复平面之间的相互关系。

12.5.1.1　S 平面到 Z 平面的映射关系

S 平面到 Z 平面的映射关系式为：

$$z = \mathrm{e}^{Ts} \tag{12-57}$$

其中，s 是复变量，也可写成 $s=\sigma+\mathrm{j}\omega$，所以 z 也是复变量，即

$$z = \mathrm{e}^{Ts} = \mathrm{e}^{T\sigma}\mathrm{e}^{\mathrm{j}\omega T} \tag{12-58}$$

极坐标形式为：

$$z = |z|\mathrm{e}^{\mathrm{j}\theta} \tag{12-59}$$

式中，$|z| = \mathrm{e}^{T\sigma}$，$\theta = \omega T$。

因此，从式（12-59）的关系可得到Z平面的稳定条件，如表12.8所示。

表12.8　Z平面稳定条件

在S平面内	系统状态	在Z平面内
$\sigma_i > 0$	系统不稳定	$\|z_i\| > 1$
$\sigma_i = 0$	系统临界稳定	$\|z_i\| = 1$
$\sigma_i < 0$	系统稳定	$\|z_i\| < 1$

由此可见，S平面的左半平面对应于Z平面以原点为圆心的单位圆内，S平面的虚轴映射为Z平面的单位圆边界，如果12.15所示。

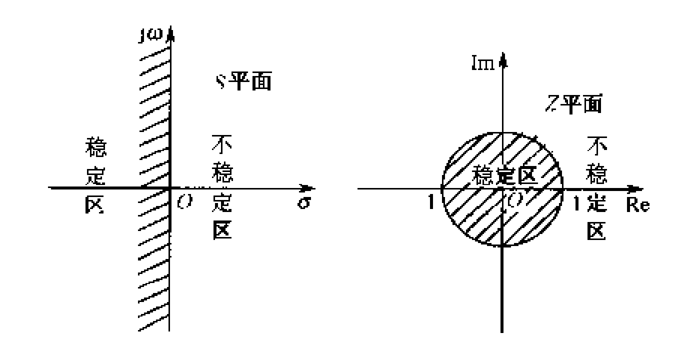


图12.15　S平面和Z平面的稳定区域

应当指出，如同分析连续系统的稳定性一样，用解特征方程根的方法来判别高阶采样系统的稳定性是很不方便的。因此，需要采用一些比较实用的判别系统稳定性的方法，其中比较常用的代数判据是劳斯判据。

12.5.1.2　离散控制系统稳定性判据

离散控制系统闭环稳定的充分条件是：闭环脉冲传递函数的全部极点均位于单位圆内。因此判断离散控制系统稳定性的最直接的方法是计算闭环特征方程的根，然后根据根的位置来确定系统的绝对稳定性。

劳斯判据是判断连续系统是否稳定的一种简单的代数判据。由于连续系统和离散系统的稳定区不同，所以在离散控制系统中不能直接应用劳斯判据，必须进行变换。基于双线性变换和劳斯判据的方法能用来判断离散控制系统的稳定性。

该方法是将离散控制系统用双线性变换将Z平面单位圆内的点映射到W平面的左半平面，然后用劳斯判据判别系统的稳定性。

复变函数双线性变换公式为：

$$z=\frac{w+1}{w-1} \tag{12-60}$$

式中W是复变量，可写成：

$$w=\sigma_w+\mathrm{j}\omega_w \tag{12-61}$$

或

$$w=\frac{z-1}{z+1} \tag{12-62}$$

这样，Z 平面单位圆内部就变换到 W 平面的左半平面，它可以从几何上和数学上加以证明，此处从略。

因此稳定性条件就变为：

$$|z|<1 \to |z|=\left|\frac{w+1}{w-1}\right|=\left|\frac{\sigma_w+\mathrm{j}\omega_w+1}{\sigma_w+\mathrm{j}\omega_w-1}\right|<1 \tag{12-63}$$

即

$$\frac{(\sigma_w+1)^2+{\omega_w}^2}{(\sigma_w-1)^2+{\omega_w}^2}<1 \tag{12-64}$$

化简得：

$$\sigma_w<0 \tag{12-65}$$

【例 12-7】　已知一个离散线性系统如图 12.16 所示，其中采样周期$T_s=1\mathrm{s}$，对象模型$G_p(s)=\dfrac{K}{s(s+1)}$，零阶保持器$G_o(s)=\dfrac{1-\mathrm{e}^{-T_s s}}{s}$，试求开环增益的稳定范围。

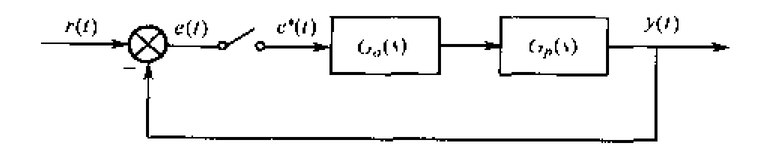


图 12.16　离散系统结构图

解：开环系统的传递函数

$$\frac{Y(s)}{E^*(s)}=G_o(s)G_p(s)=G(s)=\frac{K(1-\mathrm{e}^{-T_s s})}{s^2(s+1)}$$

对开环传递函数进行 Z 变换，并将$T_s=1\mathrm{s}$代入，得：

$$G(z)=\frac{K(0.3678z+0.2644)}{z^2-1.3678z+0.3678}$$

闭环 Z 传递函数 $T(z)$的极点是特征方程$q(z)=[1+G(z)]=0$的根，因此

$$q(z)=1+G(z)=z^2-1.3678z+0.3678+K(0.3678z+0.2644)$$

MATLAB 程序代码如下：

```
num=[0.3678, 0.2644]
den=[1, -1.3678, 0.3678]
sys=tf(num, den, -1)
%绘制根轨迹
rlocus(sys)
%选择根轨迹上的点
[k, poles]=rlocfind(sys)
```

用鼠标单击根轨迹与单位圆的交点，输出如下：

```
Transfer function:
  0.3678 z + 0.2644
---------------------
z^2 - 1.368 z + 0.3678
Sampling time: unspecified
Select a point in the graphics window
selected_point=
   0.2494 + 0.9736i
k=
    2.3874
poles=
   0.2449 + 0.9691i
   0.2449 - 0.9691i
%计算极点的模
>> abs(poles)
ans=
    0.9995
    0.9995
```

从输出结果可近似得到临界稳定的根轨迹图，根据根轨迹图可得系统稳定的 K 值范围。输出的根轨迹如图 12.17 所示。

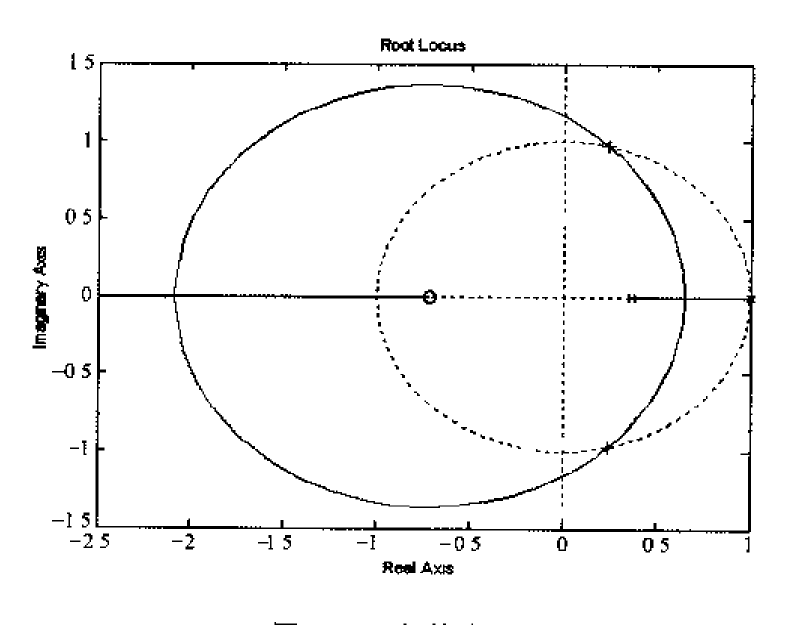


图 12.17　根轨迹图

在离输系统根轨迹图上，虚线表示的是单位圆，从根轨迹的走向以及与单位圆的交点可以大致判断系统的稳定性。

本例中，一个极点位于单位圆上，一个位于单位圆中，因此，系统在 $K=0$ 时是稳定的；随着 K 值的增大，两条根轨迹离开单位圆，系统变得不输定；随着 K 值继续增大，

虽然有一个极点落在单位圆内，但另一个极点趋向实轴的无穷远处，系统是不稳定的。所以 K 值的稳定范围是从 0 开始的一段区间。从图 12.18 上可以看出，使系统稳定的 K 值的稳定范围略为 $0<K<2.39$。

12.5.1.3　采样周期与开环增益对稳定性的影响

影响离散采样系统稳定性的因素还有系统的采样周期 T_s，根据控制系统理论，T_s 越大则采样系统的稳定性越差。

【例 12-8】 已知一个离散线性系统如图 12.18 所示，对象模型 $G_p(s)=\dfrac{2}{s(s+1)}$，$G_c(s)$ 为保持器，$r(t)$ 为单位阶跃输入，试求：

（1）当 $G_c(s)$ 为零阶保持器时，采样周期 $T_s=0.1\text{s}, 1\text{s}, 2\text{s}$ 时系统的输出。

（2）当采样周期 $T_s=1\text{s}$ 时，保持器为零阶保持器和一阶保持器时系统的输出。

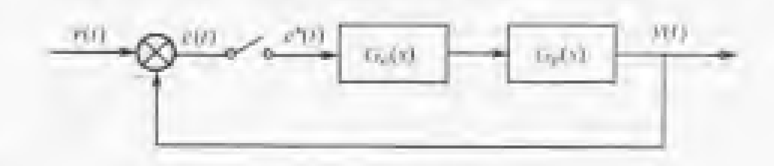


图 12.18　离散系统结构图

解：此题可利用 Simulink 中的模型离散化工具 Model Discretizer 来实现系统在不同采样周期条件下的响应。其操作步骤如下。

（1）打开 Simulink，建立系统系统模型，如图 12.19 所示。

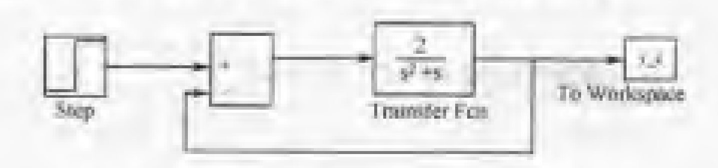


图 12.19　例 12-8 连续系统的 Simulink 仿真图

（2）选择 Model Discretizer 工具，其路径是 Tools->Control Design->Model Discretizer，如图 12.20 所示。

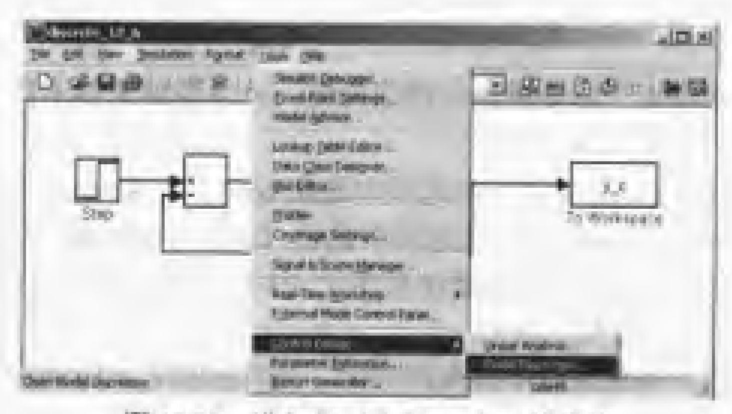


图 12.20　进入 Model Discretizer 的菜单

（3）打开 Model Discretizer 工具，如图 12.21 所示。在 Transform method 的下拉选项中选定 Z 变换的方法，这些方法有'zoh'、'foh'、'imp'、'tustin'、'prewarp'和'matched'，它们的含义在前面已做介绍；在 Sample time 中输入采样周期；在 Replace current selection with 的下拉选项中选定变换后的模型参数显示，显示的方式有 Discrete blocks （Enter parameters in s-domain）（模型框显示原连续系统的参数）、Discrete blocks （Enter parameters in z-domain）（模型框显示变换后的离散系统的参数）、Configurable subsystem （Enter parameters in s-domain）（子系统显示原连续系统的参数）和 Configurable subsystem （Enter parameters in z-domain）（子系统显示变换后的离散系统的参数）；最后单击图标 实现模型的转换，变换后的模型如图 12.22 所示。



图 12.21　Model Discretizer 界面

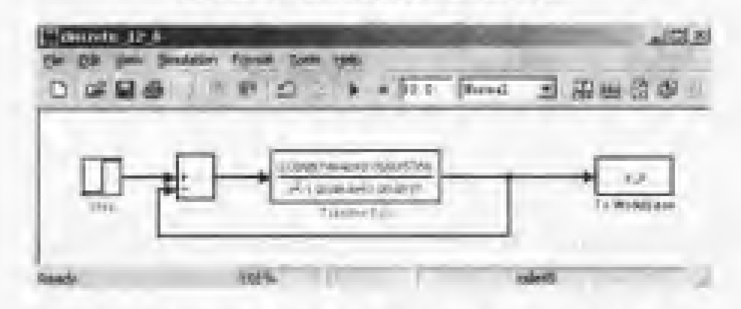


图 12.22　例 12-8 连续系统变换后的离散系统仿真图

在本例中，依次按照上述步骤进行模型转换，可方便地求得不同变换条件下系统的输出。

（4）在 Transform method 的下拉选项中选定'zoh'，在 Sample time 中输入 0.1，在 Replace current selection with 的下拉选项中选定 Discrete blocks （Enter parameters in z-domain）进行转换后并仿真，系统输出如图 12.23 所示。它是系统采用零阶保持器时，采样周期 $T_s = 0.1$s 离散后的系统输出。

用类似的方法得到系统采用零阶保持器时，采样周期 $T_s = 1$s 离散后的系统输出，如图 12.24 所示。

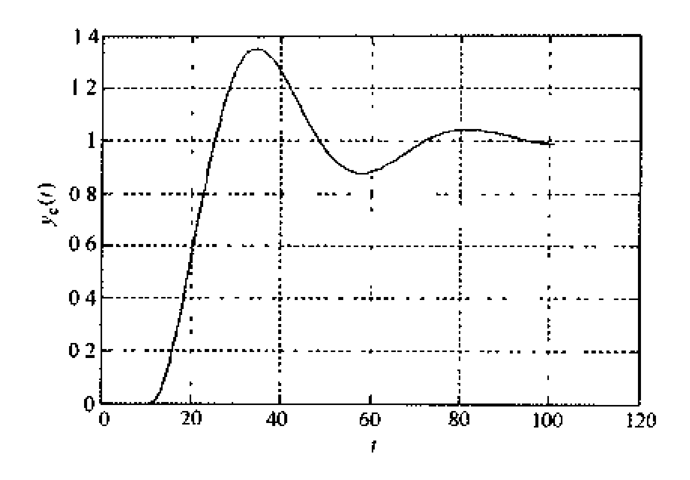


图 12.23　采用零阶保持器时，采样周期 $T_s=0.1$s 离散后的系统输出

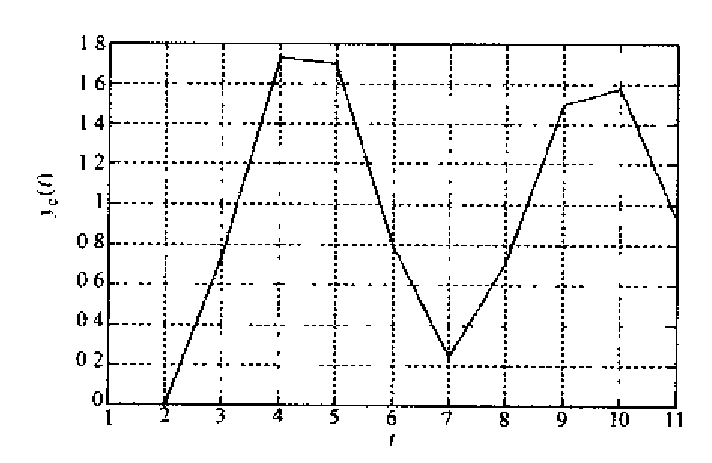


图 12.24　采用零阶保持器时，采样周期 $T_s=1$s 离散后的系统输出

当系统采用零阶保持器时，采样周期 $T_s=2$s 离散后的系统的输出如图 12.25 所示。

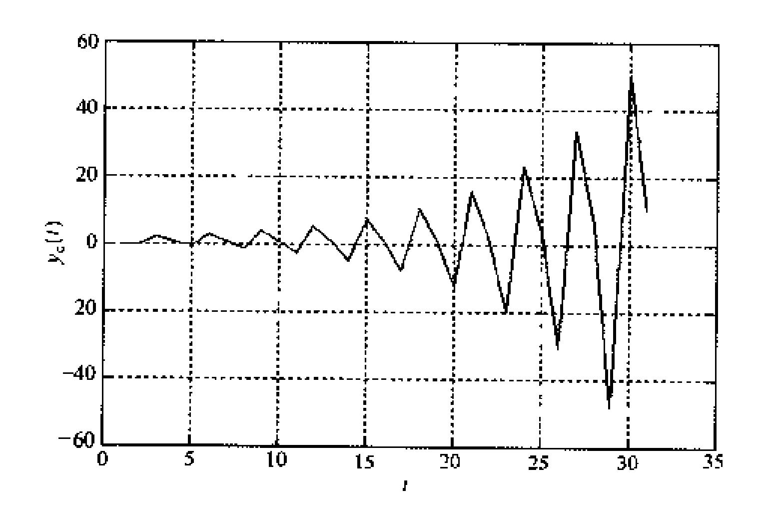


图 12.25　采用零阶保持器时，采样周期 $T_s=2$ s 离散后的系统输出

当系统采用一阶保持器时，采样周期 $T_s=1$s 离散后的系统输出如图 12.26 所示。

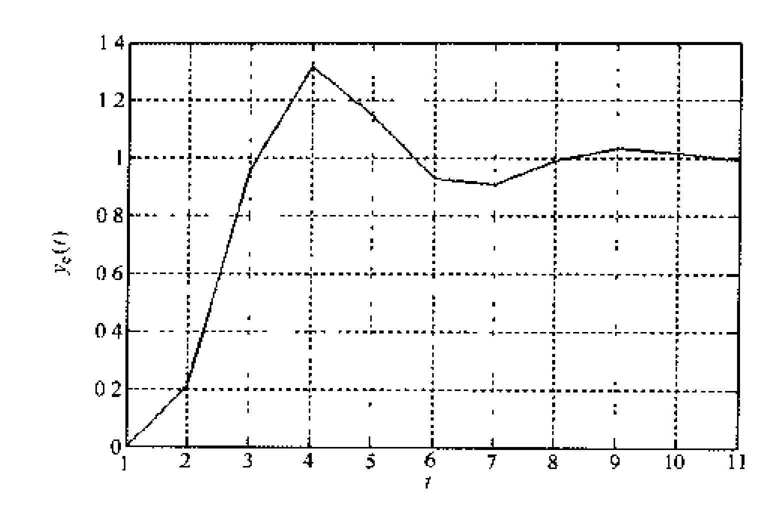


图 12.26　采用一阶保持器时，采样周期 $T_s = 1\text{s}$ 离散后的系统输出

从以上结果可以看出，在同样的保持器下，随着采样周期的增大，系统稳定性能变差；而在同一采样周期条件下，采用一阶保持器变换的系统的动态特性比采用零阶保持器变换的系统要好。

12.5.2　离散控制系统静态误差分析

在离散控制系统中，稳态误差系数的计算和线性连续系统中的类似，下面以单位反馈采样系统为例介绍系统的稳态误差系数。

设单位反馈采样系统如图 12.27 所示，其开环脉冲传递函数为 $G(z)$。

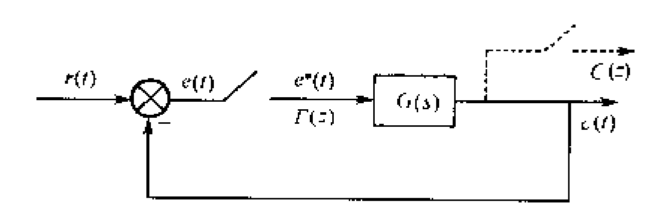


图 12.27　单位反馈采样系统

采样系统的稳态误差既可从输出信号在各采样时刻的数值 $c(nT)$ $(n = 0,1,2\cdots,\infty)$ 中得到，也可以从过渡过程曲线 $c^*(t)$ 中求取，还可以应用 Z 变换的终值定理来计算。

系统的闭环传递函数如下：

$$Y(z) = \Phi(z)R(z) \tag{12-66}$$

其中，

$$\Phi(z) = \frac{G(z)}{1+G(z)} \tag{12-67}$$

系统的误差如下：

$$E(z) = R(z) - Y(z) = R(z) - \frac{G(z)}{1+G(z)}R(z) = \frac{1}{1+G(z)}R(z) = \Phi_e(z)R(z) \tag{12-68}$$

利用 Z 变换的终值定理可求出采样瞬时的稳态误差如下：

$$e(\infty)=\lim_{t\to\infty}e^{*}(t)=\lim_{z\to1}(z-1)E(z)=\lim_{z\to1}\frac{(z-1)R(z)}{1+G(z)} \tag{12-69}$$

式（12-69）表明，系统的稳态误差与 $G(z)$及输入信号的形式有关。

由于 $z=e^{sT}$，原线性连续系统开环传递函数 $G(s)$在 $s=0$ 处极点的个数 v 作为划分系统型别的标准，可推广为将离散系统开环脉冲传递函数 $G(z)$在 $z=1$ 处极点的数目 v 作为离散系统的型别，将 $v=0, 1, 2$ 的系统分别称为 0 型、I 型、II 型离散系统。

下面是几种典型输入条件下的稳态误差。

1. 单位阶跃输入时的稳态误差

把单位阶跃函数的 Z 变换式代入式（12-69），得到单位阶跃输入时的稳态误差为：

$$e(\infty)=\lim_{z\to1}(z-1)E(z)=\lim_{z\to1}\frac{(z-1)}{1+G(z)}\cdot\frac{z}{z-1}=\frac{1}{\lim\limits_{z\to1}[1+G(z)]}=\frac{1}{k_p} \tag{12-70}$$

式中，$k_p=\lim\limits_{z\to1}[1+G(z)]$，称为静态位置误差系数。

对 0 型离散系统（没有 $z=1$ 的极点），$k_p\neq\infty$，从而 $e(\infty)\neq0$；对 I 型、II 型以上的离散系统（有一个或一个以上 z=1 的极点），$k_p=\infty$，从而 $e(\infty)=0$。

因此，在单位阶跃函数作用下，0 型离散系统在采样瞬时存在位置误差；I 型或 II 型以上的离散系统在采样瞬时没有位置误差。这与连续系统十分相似。

2. 单位斜坡输入时的稳态误差

把单位阶跃函数的 Z 变换式代入式（12-69），得到单位阶跃输入时的稳态误差为：

$$e(\infty)=\lim_{z\to1}(z-1)E(z)=\lim_{z\to1}\frac{(z-1)}{1+G(z)}\cdot\frac{Tz}{(z-1)^2}=\frac{T}{\lim\limits_{z\to1}[(z-1)G(z)]}=\frac{T}{k_v} \tag{12-71}$$

式中 $k_v=\lim\limits_{z\to1}[(z-1)G(z)]$，称为静态速度误差系数。

因为 0 型系统的 $k_v=0$，I 型系统的为有限值，II 型和 II 型以上系统的为无限值，所以有如下结论：0 型离散系统不能承受单位斜坡函数作用；I 型离散系统在单位斜坡函数作用下存在速度误差；II 型和 II 型以上离散系统在单位斜坡函数作用下不存在稳态误差。

3. 单位加速度输入时的稳态误差

把单位阶跃函数的 Z 变换式代入式（12-69），得到单位阶跃输入时的稳态误差为：

$$e(\infty)=\lim_{z\to1}(z-1)E(z)=\lim_{z\to1}\frac{(z-1)}{1+G(z)}\cdot\frac{T^2z(z+1)}{2(z-1)^3}=\frac{T^2}{\lim\limits_{z\to1}[(z-1)^2G(z)]}=\frac{T^2}{k_a} \tag{12-72}$$

式中 $k_a=\lim\limits_{z\to1}[(z-1)^2G(z)]$，称为静态加速度误差系数，当然，上式也是系统的稳态位置误差，因此也称为加速度误差。

由于 0 型及 I 型系统的 $K_a=0$，II 型系统的为常值，III 型及 III 型以上系统的 $K_a=\infty$，因此有如下结论：0 型及 I 型离散系统不能承受单位加速度函数作用；II 型离散系统在单位加速度函数作用于下存在加速度误差；只有 III 型及 III 型以上的离散系统在单位加速

度函数作用下才不存在采样瞬时的稳态位置误差。

三种类型系统的误差系数归结如表 12.9 所示。

表 12.9　三种类型系统的误差系数表

系 统 类 型	位 置 误 差	速 度 误 差	加速度误差
0 型系统	$\frac{1}{k_p}$	∞	∞
I 型系统	0	$\frac{1}{k_v}$	∞
II 型系统	0	0	$\frac{1}{k_a}$

12.5.3　离散控制系统动态特性分析

在线性连续系统中，闭环传递函数零、极点在 S 平面的分布对系统的暂态响应有非常大的影响。与此类似，采样系统的暂态响应与闭环脉冲传递函数零、极点在 Z 平面的分布也有密切的关系。

设闭环系统的脉冲传递函数为：

$$\Phi(z)=\frac{M(z)}{N(z)}=\frac{b_0z^m+b_1z^{m-1}+b_2z^{m-2}+\cdots+b_{m-1}z+b_m}{a_0z^n+a_1z^{n-1}+a_2z^{n-2}+\cdots+a_{n-1}z+a_n}\qquad(12\text{-}73)$$

式中 $m<n$。为便于分析，设其无重极点（极点分别为 $p_1, p_2, \dots, p_n$）。

采样系统的单位阶跃响应为：

$$Y(z)=\Phi(z)R(z)\qquad(12\text{-}74)$$

其中 $R(z)=z/(z-1)$，则 $Y(z)$可写成：

$$Y(z)=A_0\frac{z}{z-1}+\sum_{i=1}^{n}A_i\frac{z}{z-p_i}\qquad(12\text{-}75)$$

其中 A_i 为留数，通过 Z 反变换导出输出信号的脉冲序列 $y^*(t)$ 在技术上并无困难。

$$y(kT)=A_0 1(1)+\sum_{i=1}^{n}A_ip_i^k\qquad(12\text{-}76)$$

式中，第一项为系统输出的稳态分量，第二项为输出的暂态分量。显然，随着极点在 Z 平面位置的变化，它所对应的暂态分量也不同。

1. 实轴上闭环单极点时

设 p_i 为正实数，则 p_i 对应的暂态项为：

$$y_i^*(t)=Z^{-1}\left[A_i\frac{z}{z-p_i}\right]\qquad(12\text{-}77)$$

当 $p_i>0$ 时，

$$y_i(kT)=A_ip_i^k=A_ie^{akT}\qquad(12\text{-}78)$$

令 $a=\frac{1}{T}\ln p_i$，则其动态过程为按指数规律变化的脉冲序列。

当 $p_i<0$ 时，

$$y_i(kT)=A_i p_i^k \tag{12-79}$$

其动态过程为交替变号的双向脉冲序列。

若闭环实数极点位于右半 Z 平面，则输出动态响应形式为单向正脉冲序列。实极点位于单位圆内，脉冲序列收敛，且实极点越接近原点，收敛越快；实极点位于单位圆上，脉冲序列等幅变化；实极点位于单位圆外，脉冲序列发散。

若闭环实数极点为于左半 Z 平面，则输出动态响应形式为双向交替脉冲序列。实极点位于单位圆内，双向脉冲序列收敛；实极点位于单位圆上，双向脉冲序列等幅变化；实极点位于单位圆外，双向脉冲序列发散。

2. 闭环共轭复数极点时

设 p_h 和 $p_{h+1}=|p_h|e^{\pm j\omega_h}$ 为一对共轭复数极点，则 p_h 和 p_{h+1} 对应的暂态项为：

$$y^*(t)=Z^{-1}\left[A_k\frac{z}{z-p_k}+A_{k+1}\frac{z}{z-p_{k+1}}\right] \tag{12-80}$$

$$y_k(nT)=2|A_k|e^{anT}\cos(n\omega T+\varphi_k) \tag{12-81}$$

其中，$a=\frac{1}{T}\ln|p_k|,\ \omega=\theta_k/T,\ 0<\theta_k<\pi$。

若 $|p_h|>1$，则闭环复数极点位于 Z 平面上的单位圆外，动态响应为振荡脉冲序列。

若 $|p_h|=1$，则闭环复数极点位于 Z 平面上的单位圆上，动态响应为等幅振荡脉冲序列。

若 $|p_h|<1$，则闭环复数极点位于 Z 平面上的单位圆内，动态响应为振荡收敛脉冲序列，且模越小，即复极点越靠近原点，振荡收敛越快。

通过以上分析可以看出，闭环脉冲传递函数的极点在 Z 平面上的位置决定相应暂态分量的性质和特点。

当闭环极点位于单位圆内时，其对应的暂态分量是衰减的，极点离原点越近衰减越快。若极点位于正实轴上，暂态分量按指数衰减。一对共扼复数极点的暂态分量为振荡衰减，其角频率为 $\frac{\theta_k}{T}$。

当闭环极点位于负实轴上时，也将出现衰减振荡，其振荡角频率为 $\frac{\pi}{T}$。为了使采样系统具有较为满意的暂态响应，其 Z 传递函数的极点最好分布在单位圆内的右半部靠近原点的位置。

在线性连续系统中根据一对主导极点分析系统暂态响应的方法，也可以推广到采样系统。

综上所述，离散系统的动态特性与闭环极点的分布密切相关。当闭环实极点位于 Z 平面上左半单位圆内时，由于输出衰减脉冲交替变号，故动态过程质量很差；当闭环复极点位于左半单位圆内时，由于输出衰减高频振荡脉冲，故动态过程性能欠佳。

因此，在离散系统设计时，应把闭环极点安置在 Z 平面的右半单位圆内、且尽量靠近

极点。

在 MATLAB 中，提供了用于求离散系统时域响应和频域响应的函数，分别如表 12.10 和表 12.11 所示。

表 12.10　离散系统时域响应函数

函数名	调用格式	功能说明
dstep	dstep（dnum, dden, n） y=dstep（dnum, dden, n）	求离散系统单位阶跃响应
dimpulse	dimpulse（dnum, dden, n） y=dimpulse（dnum, dden, n）	求离散系统单位脉冲响应
dlsim	dlsim（dnum, dden, u） y=dlsim（dnum, dden, u）	求离散系统在输入 u 下的响应

注：n 为采样次数，u 为输入函数。

离散时间 LTI 系统频域分析方法主要有三种：Bode 图（开环对数幅频/相频特性曲线）法、Nyquist 曲线（开环频率特性 $G(j\omega)$ 的极坐标图）法和 Nichols 图（开环对数幅相图）法。表 12.11 列出了离散系统的频域分析函数。

表 12.11　离散系统频域响应函数

函数名	调用格式	功能说明
dbode	dbode（dnum, dden, Ts, w） [mag, phase, w]=dbode（dnum, dden, Ts, w）	离散 Bode 图
dnyquist	dnyquist（dnum, dden, Ts, w） [re, im, w]=dnyquist（dnum, dden, Ts, w）	离散 Nyquist 图
dnichols	dnichols（dnum, dden, Ts, w） [re, im, w]=dnichols（dnum, dden, Ts, w）	离散 Nichols 图
margin	margin（dsys） [Gm, Pm, Wcg, Wcp]=margin（dsys）	离散 Bode 图，显示频域性能参数

注：Ts 为采样周期；mag 为幅值向量；phase 为相交向量；Gm 为增益裕量；Pm 为相角裕量；re 为 Nyquist 图或 Nychols 图实部向量；im 为 Nyquist 图或 Nychols 图虚部向量。

【例 12-9】　若某控制系统结构如图 12.28 所示，其中 $D_1(z)=\dfrac{3.4z^{-1}-1.5z^{-2}}{1-1.6z^{-1}+0.8z^{-2}}$，$G_1$ 是零阶保持器，$G_1(s)=\dfrac{1-e^{-0.05s}}{s}$，$G_2(s)=\dfrac{0.25}{s^2+3s+2}$，采样周期 $T_s=0.05\text{s}$，试求系统开环和闭环的 Z 传递函数以及 S 传递函数，当输入为单位阶跃函数时，试求其输出。

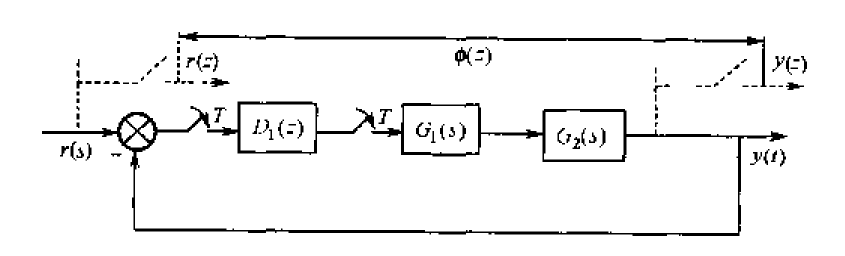


图 12.28　采样离散控制系统

解：求取系统开环和闭环 S 传递函数的程序代码如下。

```
clear
%采样周期
Ts=0.05
dnum1=[3.4, -1.5]
dden1=[1, -1.6, 0.8]
%建立 z 传递函散模型
sysd1=tf(dnum1, dden1, Ts)
%z 传递函数模型转换成 s 传递函数模型
sysc1=d2c(sysd1, 'zoh')
num2=[0.25]
den2=[1, 3, 2]
sys2=tf(num2, den2)
sysc2=sysc1*sys2
sysbc=feedback(sysc2, 1)
%提取闭环传递函数的分子和分母
[num, den]=tfdata(sysbc, 'v')
%求闭环系统特征根
p=roots(den)
```

运行结果如下：

```
%数字控制器脉冲传递函数
Transfer function:
   3.4 z - 1.5
-----------------
z^2 - 1.6 z + 0.8
Sampling time: 0.05
%数字控制器转换为 s 传递函数
Transfer function:
   55.97 s + 864.2
---------------------
s^2 + 4.463 s + 90.97
%对象部分传递函数
Transfer function:
     0.25
-------------
s^2 + 3 s + 2
%系统开环传递函数
Transfer function:
```

```
                    13.99 s + 216
---------------------------------------------
s^4 + 7.463 s^3 + 106.4 s^2 + 281.8 s + 181.9
%系统闭环传递函数
Transfer function:
                    13.99 s + 216
-------------------------------------------
s^4 + 7.463 s^3 + 106.4 s^2 + 295.8 s + 398
%求出特征方程的根，根都在 s 平面左半部，可见系统是稳定的。
p=
  -2.1667 + 9.1429i
  -2.1667 - 9.1429i
  -1.5647 + 1.4351i
  -1.5647 - 1.4351i
```

求取系统开环和闭环 Z 传递函数程序代码如下：

```
Clear
%采样周期
Ts=0.05
dnum1=[3.4, -1.5]
dden1=[1, -1.6, 0.8]
sysd1=tf(dnum1, dden1, Ts)
num2=[0.25]
den2=[1, 3, 2]
sys2=tf(num2, den2)
%s 传递函数模型转换成 z 传递函数模型
sysd2=c2d(sys2, Ts, 'zoh')
sysd=sysd1*sysd2
sysbd=feedback(sysd, 1)
%提取闭环传递函数的分子和分母
[dnum, dden]=tfdata(sysbd, 'v')
%求闭环系统特征根
pd=roots(dden)
t=0:0.05:5
%求闭环系统的单位阶跃响应
y=dstep(dnum, dden, t)
%棒图显示响应曲线
stem(t, y)
xlabel('t')
ylabel('y')
grid
```

程序执行结果如下：

```
%数字控制器脉冲传递函数
Transfer function:
   3.4 z - 1.5
----------------
z^2 - 1.6 z + 0.8
Sampling time: 0.05
%对象部分传递函数
Transfer function:
    0.25
-------------
s^2 + 3 s + 2
%对象部分传递函数转换成 z 传递函数
Transfer function:
0.0002973 z + 0.0002828
----------------------
z^2 - 1.856 z + 0.8607
Sampling time: 0.05
%系统开环 z 传递函数
Transfer function:
   0.001011 z^2 + 0.0005156 z - 0.0004242
--------------------------------------------
z^4 - 3.456 z^3 + 4.63 z^2 - 2.862 z + 0.6886
Sampling time: 0.05
%系统闭环 z 传递函数
Transfer function:
    0.001011 z^2 + 0.0005156 z - 0.0004242
---------------------------------------------
z^4 - 3.456 z^3 + 4.631 z^2 - 2.861 z + 0.6881
Sampling time: 0.05
%求出特征方程的根，根都在 z 平面单位圆内，可见系统是稳定的。
pd=
   0.8030 + 0.3935i
   0.8030 - 0.3935i
   0.9250 + 0.0697i
   0.9250 - 0.0697i
```

程序输出的图形如 12.29 所示。

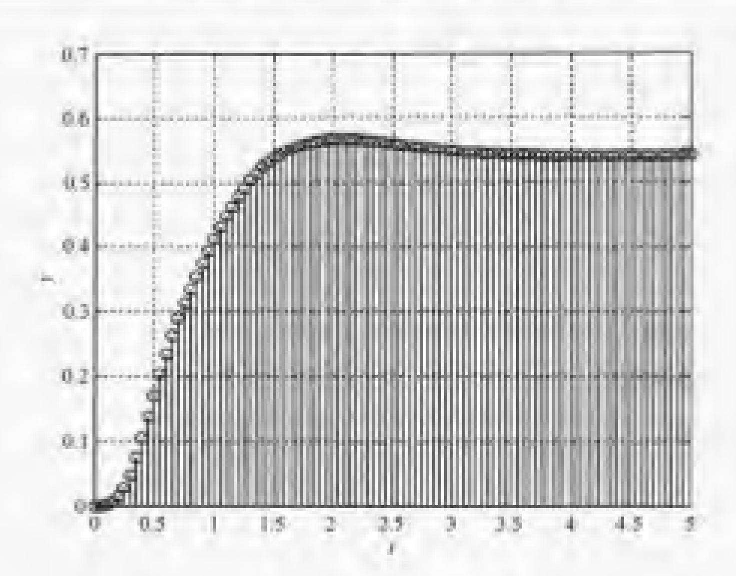


图 12.29　单位反馈采样系统

【例 12-10】　若某控制系统结构如图 12.30 所示，其中 $D_1(z)=\dfrac{3.4z^{-1}-1.5z^{-2}}{1-1.6z^{-1}+0.8z^{-2}}$，$G_1$ 是零阶保持器，$G_1(s)=\dfrac{1-e^{-0.05s}}{s}$，$G_2(s)=\dfrac{0.25}{s^2+3s+2}$，采样周期 $T_s=0.05\,s$，试求系统频率特性参数，并绘制 Bode 图、Nyquist 图和 Nichols 图。

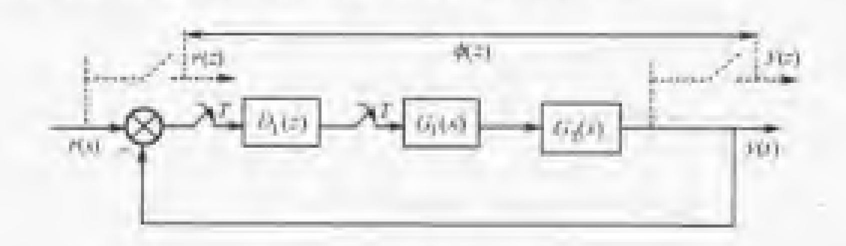


图 12.30　采样离散控制系统

解：求系统开环脉冲传递函数频率响应的程序代码如下。

```
Clear
%采样周期
Ts=0.05
dnum1=[3.4, -1.5]
dden1=[1, -1.6, 0.8]
sysd1=tf(dnum1, dden1, Ts)
num2=[0.25]
den2=[1, 3, 2]
sys2=tf(num2, den2)
sysd2=c2d(sys2, Ts, 'zoh')
sysd=sysd1*sysd2
%提取开环传递函数的零极点
```

```
[dnumc, ddenc]=tfdata(sysd, 'v')
%求系统频率特性参数
[Gm, Pm, Wcg, Wcp]=margin(sysd)
%绘制 Bode 图
margin(sysd)
grid
w=0.01:0.01:100
figure(2)
%用 nyquist 函数绘制 Nyquist 图
dnyquist(dnumc, ddenc, Ts, w)
grid
figure(3)
%用 nichols 函数绘制 Nyquist 图
nichols(sysd)
grid
```

程序执行结果如下：

```
Gm=
    10.8194
Pm=
  133.7794
Wcg=
     6.4303
Wcp=
     0.5625
```

绘制的 Bode 图如图 12.31 所示。

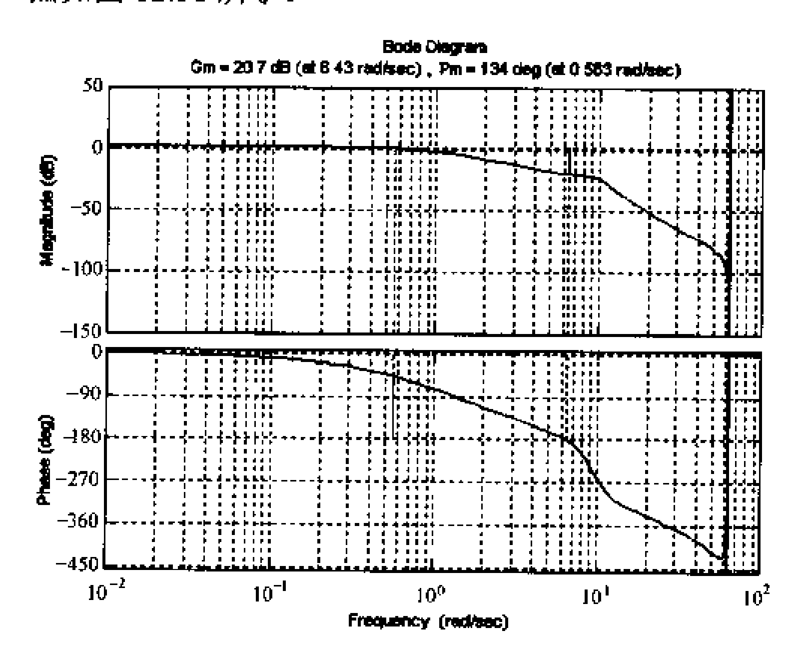


图 12.31　Bode 图及频率特性参数显示

绘制的 Nyquist 图如图 12.32 所示。

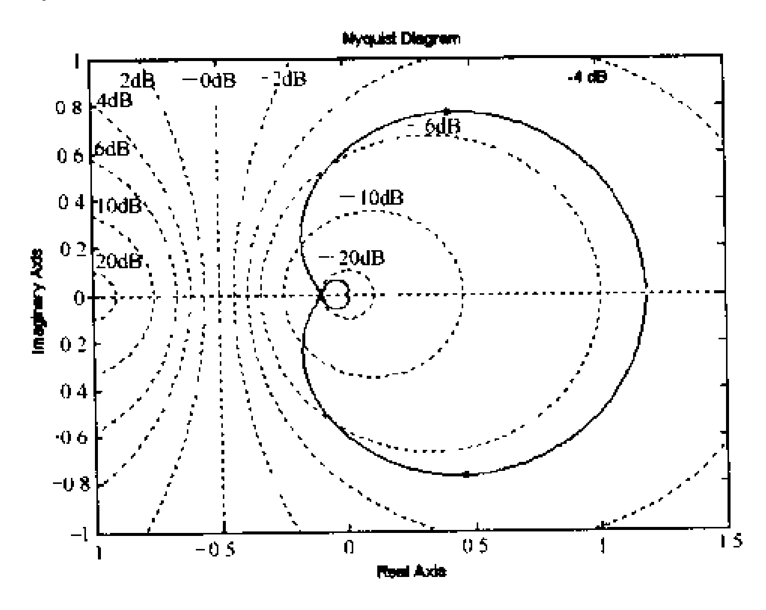


图 12.32　Nyquist 图

绘制的 Nichols 图如图 12.33 所示。

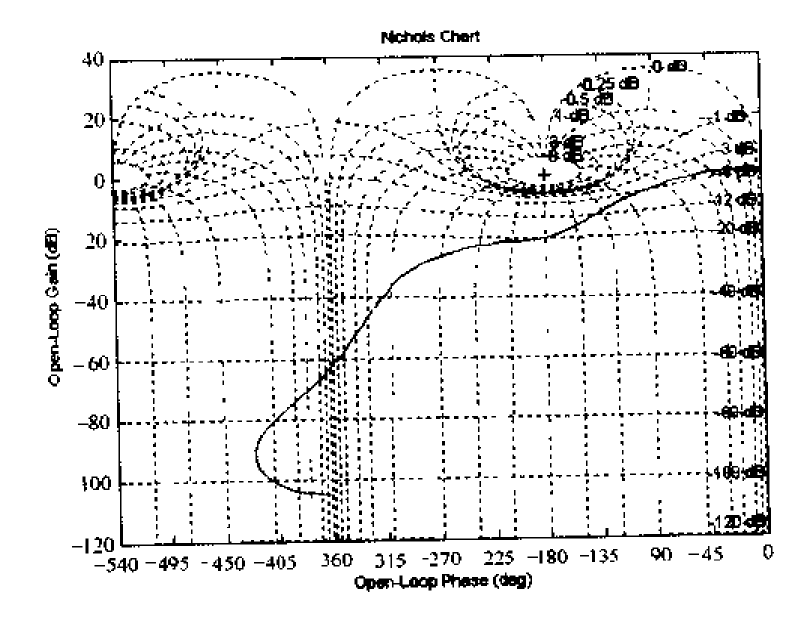


图 12.33　Nichols 图

参 考 文 献

1 胡寿松主编．自动控制原理．北京：科学出版社，2001

2 汪仁先主编．自动控制原理．北京：兵器工业出版社，1996

3 [美] Katsuhiko Ogata 著．卢伯英译．现代控制工程．北京：电子工业出版社，2003

4 何衍庆主编．控制系统分析、设计和应用．北京：化学工业出版社，2003

5 刘豹主编．现代控制理论（第二版）．北京：机械工业出版社，1988

6 刘坤主编．MATLAB 自动控制原理习题精解．北京：国防工业出版社，2004

7 薛定宇主编．反馈控制系统设计与分析．北京：清华大学出版社，2000

8 金以慧主编．过程控制．北京：清华大学出版社，1993

9 黄忠霖主编．控制系统 MATLAB 计算及仿真．北京：国防工业出版社，2001

10 [美] Richard C. Dorf 著．谢红卫译．现代控制系统．北京：高等教育出版社，2001

11 谢克明主编．自动控制原理．北京：电子工业出版社，2004

12 钱积新主编．控制系统仿真及计算机辅助设计．北京：化学工业出版社，2003

13 [美] 迪安・K・弗雷德里克著．张彦斌译．反馈控制问题．西安：西安交通大学出版社，2001

14 王正林．基于数字信号处理器 DSP 的热连轧实时仿真．冶金自动化，2004，(5)：25～28

15 蔡启仲主编．控制系统计算机辅助设计．重庆：重庆大学出版社，2003

16 杨自厚主编．自动控制原理．北京：冶金工业出版社，1990

17 王胜开．卫星通信仿真研究．2004 年电子对抗重点实验室学术年会论文集，2004

18 [美] 乔．H．周著．王健译．离散时间控制问题．西安：西安交通大学出版社，2004

19 何克忠主编．计算机控制系统．北京：清华大学出版社，1998

20 Robert H B．Modern control systems analysis and design using MATLAB．Menlo Park，California：Addision-Wesley Publishing Company，1993

21 G. Ellis．Control System Design Guide 2nd Ed，Academic Press，2000